Universidad de las Ciencias Informáticas

Facultad III

Título: Modelo matemático para la evaluación cuantitativa de la confiabilidad en la arquitectura de Cedrux.

Trabajo de diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autor: Katia Tereza Liens Pérez.

Tutor: Ing. Larisa González.

Co-Tutor: Ing. Juan José Rosales

La Habana, Cuba

Junio 2012.

Ninguna certeza existe allí donde no puede aplicarse alguna de las ciencias matemáticas.

Leonardo Da Vinci

Declaración de autoría

Declaro que soy la única autora de este trabajo y autorizo al Centro de Informatización de Gestión de Entidades de la Universidad de las Ciencias Informáticas; a hacer uso del mismo en su beneficio.

Para que así conste firmo la presente a los __ días del mes de _____ del año____.

Firma del Autor	Firma del Tutor
Katia Tereza Liens Pérez	Ing. Larisa González Álvarez

Agradecimientos

A mis padres por su ayuda, por su guía, por el ejemplo y la educación que han sembrado en mí. A ustedes debo lo que soy.

A Yilber, gracias por hacerme tu hija, y ser un padre extraordinario. Te quiero mucho.

A mi tía Virgen, mi ejemplo de honradez, humildad, laboriosidad. Quien ha sido durante toda mi vida mi tía, mi abuela, y mi madre. Te adoro.

A mis hermanos de quienes espero ser ejemplo, y los que han sido el motor impulsor de mis años de estudios. Ustedes son un tesoro muy preciado para mí.

A mi novio, mi amigo, mi compañero durante estos últimos 6 años en los que hemos crecido juntos. Por sus insuperables muestras presencia aun en la distancia. Te amo Tata eres muy especial para mí.

A mis primos Damaris y Yoba, por sus consejos, por sus llamadas, por ser mis hermanos mayores, por estar siempre cuando los necesito.

A mis suegros y cuñados por hacerme parte de su familia, por tanto cariño, y por apoyarme durante tantos años.

A mis amigos, mis compañeros, mis consejeros durante toda la universidad, todos los halagos para ellos serían pocos, Giorgyto, Ari, Martín, Diami, Daniel, Vlady. Siempre aquí en mi corazón estará el lugar tan especial que ustedes se ganaron durante estos 6 años.

A Yami, mi profesora en los tiempos más difíciles, y mi amiga incondicional. Gracias por estar desde el primer día hasta el último. Te quiero mucho.

A Lisandra, que me hizo su hija más pequeña, para no dejarme jamás sola en ninguna decisión importante.

A Yaneisi, Lazarito y su familia por ser más que mis vecinos por ser mi familia en todo momento.

A mi tutora por su esfuerzo, dedicación y apoyo durante estos meses. Muchas gracias Larisa.

Al profesor y Doctor en Ciencias Liesner Acevedo, por su valiosa e incondicional ayuda para la realización de este trabajo. Muchas gracias profesor.

A Raicel el líder de la tropa del Zulia, mi profesor, mi amigo; a Luisito, y todos mis compañeros de buenos y duros momentos en ese lejano país.

A todo el equipo de arquitectura del ERP por la inmensa ayuda que me brindaron, a Yiset y los que colaboraron con la realización de este trabajo. Le agradezco a todos los que me han ayudado a transitar por esta universidad durante estos 6 años.

Dedicatoria

A mis padres por sus años de entrega, de esmero y dedicación. Por el amor y la dulzura que siempre tienen para mí. Por confiar en mis decisiones, y ser ambos mi faro, mi estrella y la luz de mis ojos. Por darme el hermoso privilegio de tenerlos como padres. A ellos debo más de lo que podría darles en mi vida completa.

Los amo.

Resumen

La confiabilidad es un atributo de calidad relevante para los Sistemas de Planificación de Recursos Empresariales. Los arquitectos de Cedrux no cuentan en el marco de evaluación del sistema, con técnicas de evaluación dirigidas a atributos de calidad específicos. Por tal motivo, el presente trabajo tiene como objetivo definir un modelo matemático para la evaluación de la confiabilidad en la arquitectura de este sistema, que permita a los desarrolladores conocer el estado de este atributo de calidad, así como arribar a interpretaciones que permitan retroalimentar el proceso de desarrollo.

Para elaborar la propuesta se analizó el Modelo de estimación de la confiabilidad de los componentes de software mediante el aprovechamiento de los modelos arquitectónicos. A partir de las buenas prácticas de este modelo, y adaptándolas a las particularidades de Cedrux, se definió el modelo que constituye la propuesta de esta investigación. En el mismo se modela, mediante máquinas de estado, el comportamiento del componente respecto a un error determinado. Se utiliza además, el algoritmo de agrupamiento de las K-medias para agrupar las confiabilidades según sus clasificaciones (confiables, poco confiables y no confiables), a partir de las cuales se arriba al cálculo de la confiabilidad del componente.

Se realizó un análisis de los resultados obtenidos partiendo de la aplicación del modelo matemático a los subsistemas de Cedrux, mostrándose la efectividad de este mediante los indicadores determinados, y la corroboración de los arquitectos de los subsistemas evaluados.

Palabras Claves: Confiabilidad, evaluación arquitectónica, modelo matemático.

Índice de Contenido

Índice de Figuras

Índice de Tablas

Introducción

Durante la última década el ambiente competitivo en el ámbito empresarial ha mostrado un gran incremento. Las grandes compañías, empresas u organizaciones encaminan sus esfuerzos a impulsar los procesos y actividades de negocio que generen ventajas ante sus más fuertes competidores. Actualmente contar con información confiable, íntegra y oportuna puede significar una sustancial ganancia o pérdida monetaria. Esto ha generado un notable incremento de adquisiciones de paquetes de software empresariales tales como Los Sistemas de Planificación de Recursos Empresariales (ERP[1], del inglés). (Martínez, 2006)

Los sistemas del tipo ERP se han definido como sistemas globales de planificación de los recursos y de gestión de la información, que de forma estructurada pueden satisfacer la demanda de las necesidades de gestión de la empresa. Son paquetes de software que permiten a las compañías implementar, automatizar y gestionar de forma eficiente las diferentes operaciones que se presentan en estas. (Vera, 2006)

A raíz del proceso de perfeccionamiento empresarial en el que se encuentra inmerso el país, surge la necesidad de crear un Sistema de Planificación de Recursos Empresariales. Desarrollado bajo el nombre de Cedrux, este se encamina a perfeccionar el actual sistema de gestión de las empresas, basándose en los principios de independencia tecnológica, y en las particularidades propias de la economía cubana. Está diseñado para modelar e informatizar la mayoría de los procesos empresariales, así como facilitar la planificación de todos los recursos de estas instituciones.

Indudablemente una de las etapas fundamentales en el proceso de desarrollo de un ERP, es la construcción de la arquitectura. No pocos son los autores que plantean que el éxito o fracaso de un sistema está determinado por esta. (Gómez, 2005) La arquitectura afecta a todos los relacionados con el proyecto, afecta a los clientes, al gerente, al equipo de desarrollo, y al equipo de pruebas. Cada interesado[2] se preocupa por partes específicas del sistema, y esto se ve reflejado en la arquitectura.

---

[1] Enterprise Resources Planning

[2] Persona u organización que tiene influencia – directa o indirecta – o se ve influenciado por un proceso software (Ibañes, 2010)

Esta provee un lenguaje mediante el cual los interesados comprenden el sistema y se comunican para tomar decisiones importantes. (González, 2008)

Los atributos de calidad[3] constituyen elementos críticos y muy importantes que se deben considerar en la arquitectura. Estos atributos tienen un gran impacto en el desarrollo y mantenimiento del sistema influyendo notoriamente en la calidad del mismo. (Vera, 2006). Teniendo en cuenta que la calidad de software se define como el grado en el cual el software posee una combinación deseada de atributos (Barbacci, 1995), es visible la importancia de los mismos y la necesidad de su estudio en el sistema para determinar el impacto sobre este.

Si la arquitectura se encuentra deficiente en su concepto o diseño existirá un gran porciento de posibilidades de que el sistema no sea confiable, y no satisfaga las necesidades de los usuarios finales. Resulta inminente entonces, la necesidad de llevar a cabo una eficiente evaluación de esta, definiendo y estableciendo técnicas y métodos que permitan mitigar los riesgos para lograr el impacto económico esperado en el sistema.

Cuando se menciona la calidad de un sistema de gestión empresarial, la confiabilidad es un atributo que ocupa un lugar muy importante. Cabe destacar que estos sistemas tienen entre sus misiones más importantes fungir como sistemas de comunicación que distribuyen información a partir de una base de datos, siendo esta, una de las claves para obtener sus grandes ventajas. Es primordial entonces mantener la base de datos con información confiable, además de evitar la pérdida de datos que implique el incumplimiento de los objetivos iniciales definidos (Barrientos, 2012). Es fundamental evitar al máximo la ocurrencia de fallas, faltas y errores, que podrían acarrear un ERP basado en información errónea y con omisiones. (Tekinerdogan, 2005).

En una encuesta realizada a nueve arquitectos del sistema (Ver anexo 1), con el objetivo de conocer sus opiniones respecto a la evaluación de la arquitectura, y específicamente la de la confiabilidad, se detallan los resultados (figura 1) que se mencionan a continuación:

- ✓ Un 88,89 % de los arquitectos determinaron que la confiabilidad es un atributo relevante para la evaluación de la arquitectura, aunque gran parte de los

[3] Propiedades o características del sistema que importan a los interesados, y que por lo tanto afectarán el grado de satisfacción del cliente. (Castillo, 2009)

arquitectos no realizan las evaluaciones arquitectónicas, solo en un 33,33 % de los encuestados la respuesta es afirmativa.

- ✓ Se demostró escasos conocimientos de los métodos y técnicas a aplicar para lograr el éxito de la evaluación de la confiabilidad, arrojando que solo un 11,11 % conoce algún método de evaluación.
- ✓ Solo un 22,22 % de los arquitectos creen en la factibilidad de la evaluación cualitativa de la confiabilidad, mientras que un 77,78 % opinan que mediante técnicas cuantitativas se puede agilizar y retribuir el proceso de desarrollo del software.
- ✓ Un 66,67 % considera que mediante la evaluación de la arquitectura puede obtener resultados cuantitativos que reflejen cuán satisfactorio es el diseño arquitectónico.

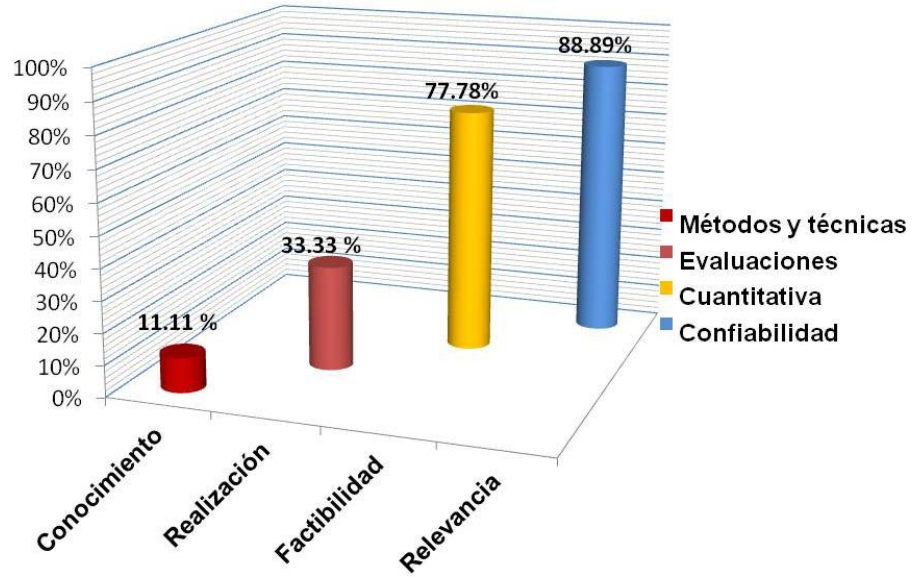

Figura 1. Resultados encuesta realizada a arquitectos de Cedrux.

En la encuesta se evidencia la alta prioridad asignada por los arquitectos a la confiabilidad para el sistema, y la necesidad de una evaluación cuantitativa que refleje resultados que permitan agilizar el proceso de desarrollo. Sin embargo, el equipo de arquitectura de Cedrux, no tiene definidas en su marco de evaluación técnicas dirigidas a atributos de calidad específicos, en este caso la confiabilidad. Si bien es cierto que el Centro Nacional de Calidad Software (CALISOFT) propone un catálogo de métricas para la evaluación de los atributos de calidad, en las pruebas actuales no se evalúan los atributos de calidad según la ISO 9126. Solo se tienen en cuenta en la reunión de cierre de las pruebas, la cantidad de No Conformidades encontradas al producto, dando una referencia del estado del producto al final de la liberación.

(Góngora, 2011) Este catálogo está dirigido a métricas desde el punto de vista de productos, por lo que incluyen variables propias de la implementación. Esto provoca que el equipo de desarrollo no obtenga mediante estas pruebas, información especificada de la arquitectura que permita retroalimentar el proceso de desarrollo.
Por lo anterior salta a la vista la necesidad de definir una técnica que guíe el proceso de evaluación de la confiabilidad en la arquitectura. Esta debe facilitar además, la obtención de resultados que ofrezcan una visión del estado o comportamiento de la confiabilidad en el sistema. El equipo de arquitectos requiere de resultados cuantitativos, que permitan la comparación para detallar los errores arquitectónicos; la posibilidad de identificarlos y corregirlos, así como prevenirlos en la medida en que sea posible a partir de la experiencia de lo ya identificado.

Por lo anteriormente expuesto, el **problema a resolver** es el siguiente: ¿Cómo evaluar cuantitativamente la confiabilidad en la arquitectura de Cedrux, de forma que permita retroalimentar el proceso de desarrollo?
Objeto de estudio: Evaluación de la confiabilidad en la arquitectura de software.

Campo de acción: Evaluación cuantitativa de la confiabilidad en la arquitectura de software.

Objetivo general: Definir un modelo matemático, para la evaluación cuantitativa de la confiabilidad en la arquitectura de Cedrux, que permita retroalimentar el proceso de desarrollo.

Objetivos específicos:

- ✓ Realizar el marco teórico de la investigación para sustentar los conceptos y el modelo de la propuesta.
- ✓ Definir un modelo matemático que permitan evaluar la confiabilidad de un componente de la arquitectura de Cedrux, y retroalimentar el proceso de desarrollo.
- ✓ Validar la propuesta del modelo.

Tareas de investigación

- ✓ Desarrollo de la fundamentación teórica del tema donde se aborde la evaluación cuantitativa de la arquitectura de un sistema informático.
- ✓ Análisis y evaluación de la información obtenida de la investigación, adoptando una posición.

- ✓ Identificación de un modelo matemático que, atendiendo a sus características, pueda ser utilizado en la evaluación de la confiabilidad de la arquitectura.
- ✓ Identificación de las buenas prácticas del modelo identificado y aquellas que no se adecuen al sistema
- ✓ Modificación del modelo seleccionado a las particularidades del sistema Cedrux, para su utilización en la evaluación de su arquitectura.
- ✓ Aplicación del modelo definido a los subsistemas de Cedrux.
- ✓ Verificación del resultado del modelo definido mediante la corroboración de los arquitectos del sistema.

Idea a defender: Si se define un modelo para la evaluación de la confiabilidad en la arquitectura de Cedrux, se podrán corregir errores retribuyendo el proceso de desarrollo del software.

Métodos

Histórico lógico: Utilizado para el análisis y la obtención de fundamentos de la arquitectura de software, así como la evaluación cuantitativa de la confiabilidad mediante modelos matemáticos, logrando un mayor entendimiento de su evolución y aplicación actual.

Hipotético-deductivo: Para la solución al problema se analizó un modelo matemático existente para la evaluación de la arquitectura. Teniendo en cuenta la idea a defender, este fue modificado y particularizado a partir de deducciones lógicas, para obtener un nuevo modelo adaptado a Cedrux, que retribuya el proceso de desarrollo. Esto posteriormente sería sometido a verificaciones empíricas.

Analítico-Sintético: Utilizado para el estudio de los factores que condicionan la evaluación cuantitativa de la arquitectura, así como las relaciones e interacciones que existen entre estos factores.

Entrevista: Para obtener información mediante los especialistas del comportamiento del componente a partir de un error detectado, así como corroborar la solución del modelo.

Encuesta: Para la determinación de los errores más dañinos al sistema, y la clasificación que más se adapte Cedrux.

Experimentación: Para poner en práctica el modelo matemático y la obtención de los resultados tras su aplicación en el sistema.

Posibles resultados: Modelo matemático para la evaluación de la confiabilidad de la arquitectura de Cedrux que permita retroalimentar el proceso de desarrollo.

Estructuración del contenido

Capítulo 1: Fundamentación Teórica

Comprende el estado del arte sobre el tema tratado, haciendo énfasis en la evaluación de la arquitectura, y en la importancia de esta. Se expone el marco conceptual sobre el que gira la propuesta de solución. Se realiza un análisis de las técnicas cuantitativas, y en ellas los modelos matemáticos para la evaluación de la confiabilidad, determinando el modelo que va a servir de guía para la definición de un nuevo modelo de evaluación.

Capítulo 2: Desarrollo de la solución propuesta

En este capítulo se realizará el análisis de la solución propuesta, especificando las buenas prácticas del modelo estudiado, así como aquellas que no se adecuan al sistema. Se describe el modelo definido para la evaluación cuantitativa de la confiabilidad de Cedrux, detallando las ventajas de este para el equipo de desarrollo.

Capítulo 3: Validación de la solución propuesta

Este capítulo muestra los resultados obtenidos luego de la aplicación del modelo matemático a los subsistemas de Cedrux. Se analiza mediante indicadores la retribución del modelo al proceso de desarrollo. También se muestran en él, los criterios de los arquitectos del sistema sobre la validez de la propuesta y su puesta en práctica.

Capítulo I: Fundamentación Teórica

El primer capítulo enuncia criterios y conceptos de diferentes autores que se consideran de importancia para la fundamentación y comprensión del trabajo realizado. Se hace referencia entre ellos a la calidad, a la arquitectura de software y su importancia durante el proceso de desarrollo. Aborda la calidad arquitectónica, los atributos y modelos de calidad como la base en la búsqueda de un mejor sistema de forma global. Se enuncia el concepto de confiabilidad definido por el modelo de calidad acogido en la investigación. Además, se expone la importancia de evaluación de la arquitectura, enfatizando en las técnicas de evaluación cuantitativas. Por último, se realiza un estudio de dos modelos matemáticos para la evaluación de la confiabilidad, y se concreta el que va a servir de guía para el modelo a definir en la investigación. Este modelo será el encargado de detectar posibles riesgos en el sistema y brindar recomendaciones para contenerlos.

1.1 Calidad del software

Uno de los principales problemas que se afronta hoy en día durante la realización de un sistema es la calidad del software. A partir de la década del 70 y hasta la actualidad, este tema es motivo de preocupación para especialistas, ingenieros, investigadores y comercializadores de software. Las investigaciones realizadas se han encaminado principalmente hacia dos vertientes:

1. ¿Cómo obtener un software con calidad?
2. ¿Cómo evaluar la calidad del software?

El estudio de ambas interrogantes puede arrojar amplias respuestas, pero están estrechamente ligadas con el concepto de la calidad del software. Muchas organizaciones y personalidades de diferentes épocas reconocidas mundialmente, han expresado definiciones para este término, algunas de las cuales se mencionan a continuación:

“Es el conjunto de cualidades que caracterizan al software y que determinan su utilidad y existencia”. (Fernández, 1995)

Pressman la define como “La concordancia con los requisitos funcionales y de rendimiento establecidos con los estándares de desarrollo explícitamente

documentados, y con las características implícitas que se espera de todo software desarrollado de forma profesional". (Camacho, 2004)

Según la ISO/IEC[4] define a la calidad de software como la totalidad de rasgos y atributos de un producto de software que le apoyan en su capacidad de satisfacer sus necesidades explícitas o implícitas. (ISO/IEC, 1998)

La IEEE plantea que, "La calidad del software es el grado con el que un sistema, componente o proceso cumple los requisitos especificados y las necesidades o expectativas del cliente o usuario" (IEEE, 1990).

Esta última ha sido la adoptada por la investigación debido a que a pesar de que todas las mencionadas varían según el criterio de cada autor, es notorio destacar que las definiciones de calidad de software plasmadas anteriormente conllevan al planteamiento de que un software de calidad debe satisfacer los requisitos dados por el usuario, y esto queda perfectamente descrito en la definición de la IEEE.

La calidad del software puede medirse después de elaborado el producto. Pero esto puede resultar muy costoso si se detectan problemas derivados de imperfecciones en el diseño. Por eso es importancia de una buena calidad de la arquitectura, que no de margen a posibles errores futuros. (Camacho, 2004). Es ahí donde entran a desempeñar un papel importante los modelos de calidad.

1.1.1 Modelos de calidad

Los Modelos de Calidad son aquellos documentos que integran la mayor parte de las mejores prácticas, proponen temas de administración en los que cada organización debe hacer énfasis, integran diferentes prácticas dirigidas a los procesos claves, y permiten medir los avances en la calidad. (Scalone, 2006) Para cualquier organización dedicada a la investigación, producción y comercialización de software, es de primordial importancia tener en cuenta estos modelos, ya que permiten a las empresas de software realizar tareas y funciones teniendo en cuenta la calidad.

Dado que los factores que afectan a la calidad del software no cambian, resulta útil el estudio de los modelos de calidad que han sido propuestos en este sentido. Estos basan la descomposición de la calidad global en características de calidad. (Pressman, 2002) Es interesante destacar que la organización y descomposición de

[4]Intenational Standart Organitation

los atributos de calidad, ha permitido el establecimiento de modelos específicos para efectos de la evaluación de la calidad arquitectónica. (Camacho, 2004).

Modelo de Calidad McCall

El modelo de McCall fue el primero en ser presentado en 1977, y se focaliza en el producto final identificando atributos claves. Desde el punto de vista del usuario estos atributos se denominan factores de calidad y son normalmente atributos externos, pero también se incluyen algunos atributos posiblemente internos.
Tiene tres perspectivas principales para definir e identificar la calidad de un producto:

- ✓ La revisión del producto (la capacidad de sufrir cambios).
- ✓ La transición del producto (la capacidad de adaptación a los nuevos entornos).
- ✓ Operaciones de productos (sus características de operación).

Cada perspectiva se descompone en una serie de factores que determinan la calidad de cada uno de ellos. Este modelo tiene entre sus principales desventajas:

- ✓ Fijo, sin posibilidades de ser modificado.
- ✓ Métricas de carácter general, difíciles de aplicar a la arquitectura de software como producto.

McCall define la Confiabilidad como “La habilidad del producto de responder ante situaciones no esperadas”. Este delimita además este atributo en tres subcaracterísticas:

- ✓ Consistencia.
- ✓ Exactitud.
- ✓ Tolerancia a fallas.

Modelo de Calidad FURPS

El modelo FURPS propuesto por Robert Grady y Heweltt Packard Co (HP) en 1992, tiene 2 categorías de requisitos, las cuales son:

1- Requisitos funcionales (F).
2- Requisitos no funcionales (URPS).

En este modelo se desarrollan un conjunto de factores de calidad de software, bajo el acrónimo de FURPS: funcionalidad (Functionality), usabilidad (Usability), confiabilidad (Reliability), desempeño (Performance) y capacidad de soporte (Supportability).
Delimita la confiabilidad en 5 subcaracterísticas fundamentales:

- ✓ Frecuencia y severidad de las fallas

- ✓ Exactitud de las salidas
- ✓ Tiempo medio de fallos
- ✓ Capacidad de recuperación ante fallas
- ✓ Capacidad de predicción

Modelo ISO/IEC 9126

Durante muchos años se buscó en la Ingeniería de Software un modelo único para expresar calidad, este permitiría la ventaja de comparar productos entre sí. La Organización Internacional para la Estandarización (ISO), ha emitido algunas normas que definen un modelo de calidad del software en varios contextos de uso. Entre ellos una variante del modelo de McCall fue propuesta como estándar internacional para la medición de la calidad del software, ISO 9126 Software Product Evaluation: Quality Characteristics and Guide lines for their, ese es el nombre formal. El estándar ISO/IEC 9126, ha sido desarrollado para identificar los atributos de calidad claves para un producto de software (Pressman, 2002). Este modelo es completamente jerárquico, cada característica tiene un juego de características subalternas. (Panovski, 2008)
El modelo se basa en los atributos de calidad que se relacionan directamente con la arquitectura: funcionalidad, confiabilidad, eficiencia, mantenibilidad y portabilidad. Presenta importantes ventajas como:

- ✓ Es el modelo que más elementos aporta al análisis de la calidad arquitectónica.
- ✓ Facilita la comprensión de elementos que debe tener una arquitectura de Software (AS[5]), para lograr valores adecuados de cada atributo de calidad.
- ✓ Basado en los atributos de calidad que se relacionan directamente con la arquitectura.

ISO/IEC 9126 define la confiabilidad como “La capacidad del producto de software para mantener un nivel de ejecución especificado cuando se usa bajo las condiciones especificadas. Lo divide en tres subcaracterísticas:

- ✓ Madurez
- ✓ Tolerancia ante fallos
- ✓ Recuperabilidad

Consideraciones generales de los modelos de calidad

[5]Arquitectura de Software

Según (Barbacci, 1995), la confiabilidad en un sistema de software “Es la medida de la habilidad de un sistema a mantenerse operativo a lo largo del tiempo”. Durante la investigación se dividió este concepto en tres factores que tributan a la conceptualización del mismo. En la tabla 1 se muestra los resultados de un estudio de las subcaracterísticas de los modelos de calidad que contribuyen a cada uno de estos factores.

Tabla 1. Comparación de los modelos de calidad según las subcaracterísticas que determinan los factores de la definición de confiabilidad.

Factores	**Subcaracterísticas definida para el factor según el modelo de calidad**		
	McCall	**FURPS**	**ISO/IEC 9126**
Evadir colapso del sistema.			Madurez
Conservar la ejecución según lo definido.	Tolerancia a fallas		Tolerancia ante fallos
Posibilidad de recuperación		Recuperabilidad	Recuperabilidad

Luego de los análisis de cada uno de los modelos de calidad mencionados anteriormente, se observa que de manera general todos abarcan los factores de calidad que influyen directamente en la calidad del software, funcionalidad, eficiencia, mantenibilidad, confiabilidad y portabilidad. Sin embargo, las ventajas del ISO/IEC 9126 saltan a la vista, este modelo abarca atributos importantes y decisivos para la calidad arquitectónica, se divide en las subcaracterísticas que comprenden la conceptualización de confiabilidad. Está respaldado por la Organización Internacional para la Estandarización, que se dedica a estudiar y proponer estándares de calidad, y la Oficina Cubana de Normalización se acogió a este modelo para la producción de sistemas de software. Lo anterior lo convirtió en el principal candidato, por lo que se definió para la investigación el Modelo ISO/IEC 9126.

1.2 Arquitectura de software

Los primeros estudios formales de arquitectura de software, fueron encabezados por David Garland y Mary Shaw en la década de los 90 del siglo pasado. Estos reseñaron escuetamente la prehistoria de la especialidad. Sus párrafos han sido reutilizados una y otra vez en la literatura, haciendo referencia al surgimiento del concepto, a manos del reconocido Edsger Dijkstra en el año 1960. (Reynoso, 2004) La Arquitectura de

Software es una disciplina que se encuentra contenida en el ámbito del proceso de desarrollo de software. Representa una vista de abstracción del sistema como un todo, y el trabajo vinculado a esta se manifiesta durante todo el ciclo de vida del proyecto.

Definiciones

No es novedad que actualmente en la literatura se pueden encontrar circulando numerosas definiciones del término “Arquitectura de Software”, es por tanto evidente que no existe aún la conceptualización definitiva. A continuación se muestran algunos conceptos reconocidos.

“La Arquitectura de software es, a grandes rasgos, una vista del sistema que incluye los componentes principales del mismo, la conducta de esos componentes según se le percibe desde el resto del sistema, y las formas en que los componentes interactúan y se coordinan para alcanzar la misión del sistema. La vista arquitectónica es una vista abstracta, aportando el más alto nivel de comprensión y la supresión o diferimiento del detalle inherente a la mayor parte de las abstracciones”. (Clements, 1996)

“La arquitectura comprende un conjunto de decisiones significativas con respecto a la organización de un sistema de software, incluyendo la selección de los elementos estructurales y sus interfaces que componen al sistema, el comportamiento y las especificaciones de la colaboración entre esos componentes, la interdependencia entre los elementos estructurales y de comportamiento en sistemas de gran envergadura, y el estilo arquitectónico que guía esta organización”. (Meier, 2008)

“La arquitectura de software de un programa o sistema de computación, es la(s) estructura(s) del sistema que comprende los componentes del software, las propiedades visibles de esos componentes, y las relaciones entre ellos.” (Bass, 2003)

Si bien es cierto que ninguno de los conceptos anteriores es respaldado unánimemente por la totalidad de los arquitectos, existe opinión general de que ella se refiere a la estructura a grandes rasgos del sistema, estructura que consiste en componentes y relaciones entre ellos. La definición de AS que se toma como referencia en la investigación es la que brinda el documento de IEEE[6] Std 1471-2000, adoptada también por Microsoft. Esta goza de un amplio prestigio internacional, resume y esclarece los planteamientos anteriores quedando conceptualizada de la siguiente manera:

[6] Institute of Electrical and Electronic Engineers

"La Arquitectura de Software es la organización fundamental de un sistema encarnada en sus componentes, las relaciones entre ellos y el ambiente, y los principios que orientan su diseño y evolución". (Reynoso, 2004)

Luego del análisis de las definiciones se puede resumir que la arquitectura de software constituye el esquema de más alto nivel de la estructura de un sistema.

Es válido mencionar la definición de componente de software, teniendo en cuenta el posterior uso del término y su importancia dentro del concepto de Arquitectura de Software. Para ello se toma la definición adoptada y publicada por el Software Engineering Institute (SEI): "Un componente es un fragmento de un sistema de software que puede ser ensamblado con otros fragmentos para formar piezas más grandes o aplicaciones completas."

Importancia de la arquitectura

La necesidad del manejo de la arquitectura surge precisamente con los sistemas de mediana o gran envergadura. En la medida que los sistemas de software crecen en complejidad, bien sea por número de requisitos o por el impacto de los mismos, se hace necesario establecer medios para el manejo de esta complejidad. En general, la técnica es descomponer el sistema en piezas que agrupan aspectos específicos del mismo, producto de un proceso de abstracción, y que al organizarse de cierta manera constituyen la base de la solución de un problema en particular (Camacho, 2004). A continuación se muestran algunas razones que reflejan la importancia de la arquitectura para cualquier sistema de software (González, 2008).

1. *Comunicación entre interesados:* Al ser la arquitectura de software la representación de una abstracción de alto nivel de un sistema, permite que la mayoría de los interesados tomen decisiones y haya un entendimiento mutuo.
2. *Decisiones de diseño:* La arquitectura de software permite hacer diseños que se acomoden a las necesidades del sistema, a los requisitos funcionales y a los no funcionales.
3. *Abstracción transferible de un sistema:* La arquitectura de software muestra la estructura de todo un sistema. Esto permite que se haga un re-uso de los diferentes diseños en sistemas con requisitos similares.

La importancia de la arquitectura se hace evidente además, en la medida en que esta se tome como un pilar durante proceso de desarrollo para lograr calidad requerida del software.

1.3 Calidad arquitectónica

El desarrollo de formas sistemáticas para relacionar atributos de calidad de un sistema a su arquitectura, provee una base para la toma de decisiones objetivas sobre acuerdos de diseño, y permite a los ingenieros realizar predicciones razonablemente exactas sobre los atributos del sistema que son libres de prejuicios y asunciones no triviales. El objetivo de fondo es lograr la habilidad de evaluar tanto cualitativa como cuantitativamente y llegar a acuerdos entre múltiples atributos de calidad para alcanzar un mejor sistema de forma global. (Barbacci, 1995)

Atributos de calidad

Los atributos de calidad son aspectos del sistema, que en general, no afectan directamente la funcionalidad necesitada, sino que definen la calidad y las características que el sistema debe soportar (Barbacci, 1995). Otros autores plantean que tales atributos son requisitos adicionales del sistema, y que referencian características que este debe satisfacer diferentes de los requisitos funcionales (Kazman, 2001).

La clasificación de los atributos de calidad está establecida en dos categorías:

- ✓ Observables vía de ejecución: Son aquellos atributos que se determinan del comportamiento del sistema en tiempo de ejecución.

- ✓ No observables vía de ejecución: Son aquellos atributos que se establecen durante el desarrollo del sistema. (Bass, 1998)

A continuación, en la tabla 2, se muestran los atributos pertenecientes a cada una de las clasificaciones.

Tabla 2. Atributos de calidad observables y no observables vía ejecución. (Vera, 2006)

Atributos de calidad	
Observables vía de ejecución	**No observables vía de ejecución**
Disponibilidad	Configurabilidad
Confidencialidad	Integrabilidad
Funcionalidad	Integridad
Desempeño	Interoperabilidad
Confiabilidad	Modificabilidad
Seguridad externa	Mantenibilidad

Seguridad interna	Portabilidad
	Reusabilidad
	Capacidad de Prueba
	Escalabilidad

Es importante destacar que la calidad del sistema debe ser considerada en todas las fases del diseño, pero los atributos de calidad se manifiestan de maneras distintas a lo largo de estas fases. Realizar adecuadamente una identificación de la importancia de atributos de calidad, tanto de los observables como los no observables, propiciará un correcto desempeño del sistema respecto a temas de calidad, a la vez de dotar al producto con ciertas características, que determinarán en parte, el grado de aceptación del mismo, siendo estos atributos independientes de los requisitos funcionales. (Milán, 2011)

Para alcanzar un atributo de calidad específico, es necesario tomar decisiones de diseño arquitectónico que requieren un pequeño conocimiento de funcionalidad. No tener en cuenta algún requisito no funcional puede traer consigo una pérdida significativa en cuanto a costo y tiempo, a diferencia de los requisitos funcionales que con solo unas líneas de código pueden ser corregidos o modificados. Actualmente se considera que los requisitos no funcionales deben guiar la definición de la arquitectura del sistema tanto como los funcionales.

Especificidades para el atributo de calidad confiabilidad según el modelo de calidad ISO 9126.

El modelo ISO/IEC 9126 define la confiabilidad como “La capacidad del producto de software para mantener un nivel de ejecución especificado cuando se usa bajo las condiciones especificadas”

Este modelo plantea que el software no sufre desgaste ni envejecimiento. Las limitaciones en la confiabilidad son debidas a los fallos en los requisitos, el diseño y la implementación. Los fallos totales ocurridos debido a estos errores dependen de la manera en que se utilice el producto del software y las opciones del programa seleccionado, y no del tiempo de uso transcurrido.

En el documento, ISO/IEC 9126-1: 2005 la definición de confiabilidad se ha ampliado para mantener su nivel de ejecución quedando de la siguiente manera:

- ✓ **Madurez:** Capacidad del producto de software de evitar un fallo total como resultado de haberse producido un fallo del software.
- ✓ **Tolerancia ante fallos:** Capacidad del producto de software de mantener un nivel de ejecución o desempeño especificado en caso de fallos del software o de infracción de su interfaz especificada.

Un nivel de ejecución especificado puede incluir la falta de capacidad de seguridad ante errores.

- ✓ **Recuperabilidad:** Capacidad del producto de software de restablecer un nivel de ejecución especificado, y recuperar los datos directamente afectados en caso de fallo total.

Después de un fallo total, un producto del software a veces estará desactivado por un cierto período de tiempo, cuyo plazo se evalúa a partir de su recuperabilidad.

El término “Conformidad con la confiabilidad” es definido como la capacidad del producto de software para adherirse a las normas que se le apliquen, convenciones, regulaciones, leyes y las prescripciones similares, relativas a la confiabilidad. (Oficina Nacional de Normalización, 2005)

La confiabilidad dentro del sistema es un pilar para el buen funcionamiento del mismo, la carencia de este atributo puede significar un caos para la empresa u organización. De ahí la importancia de revelar y corregir posibles errores que puedan afectar la confiabilidad de un producto de software.

1.4 Evaluación de la arquitectura de software

La evaluación es un estudio de factibilidad que pretende detectar posibles riesgos, así como buscar recomendaciones para contenerlos. Los primeros esfuerzos en realizar evaluaciones a arquitecturas de software utilizando un proceso estructurado y definido, fueron descritos en un trabajo seminal hecho por Parnas y Weiss en el año de 1985. En él se propone utilizar revisiones de diseño activas, que consisten en detectar errores e inconsistencias en el diseño, por ejemplo aquellos que no fueron detectados en la fase de requisitos. (Carrascoso, 2009)

Según (Kazman, 2001) el primer paso para la evaluación de una arquitectura es conocer qué es lo que se quiere evaluar. De esta forma, es posible establecer la base para la evaluación, puesto que la intención es saber qué se puede evaluar y qué no. Resulta interesante el estudio de la evaluación de una arquitectura: si las decisiones que se toman sobre la misma determinan los atributos de calidad del sistema, es

entonces posible evaluar las decisiones de tipo arquitectónico con respecto al impacto sobre estos atributos.

Características de una Evaluación de Arquitectura

- ✓ Es uno de los principales puntos de evaluación dentro del proyecto, ya que errores en ella, pueden traer que el proyecto fracase.
- ✓ Puede ser realizada por personal Interno o Externo al proyecto, aunque lo más interesante es que sea realizada por personas Externas (Mentores o Arquitectos del Área). (Brey, 2005)

Importancia de la evaluación de la arquitectura

Después de que el sistema de software está construido, es posible determinar si cuenta con la calidad esperada verificando si este cumplió o no con los atributos de calidad que fueron especificados en los requisitos no funcionales. El atraso de un gran número de proyectos en la actualidad, se debe a que presentan problemas con la arquitectura, lo que provoca que excedan el costo de construcción. Como una buena práctica de ingeniería lo recomendable es llevar acabo evaluaciones a la arquitectura en el momento oportuno, para reducir estos riesgos.

Es importante tener en cuenta que realizar una evaluación de la arquitectura, es la manera más económica de evitar desastres. Esta práctica permite analizar y evaluar la calidad exigida por los usuarios, así como las decisiones del diseño**.** A grandes rasgos entre los principales beneficios de realizar una evaluación arquitectónica se encuentran los siguientes (Carrascoso, 2009):

- ✓ Financieros.
- ✓ Reúne a los interesados.
- ✓ Fuerza una articulación en las metas específicas de calidad.
- ✓ Fuerza una mejora a la documentación de la arquitectura.
- ✓ Mejora la arquitectura.
- ✓ Detección temprana de problemas.
- ✓ Valida los requisitos y los prioriza.
- ✓ Recolecta los fundamentos y las decisiones arquitectónicas tomadas.

Cuanto más temprano se encuentre un problema en un proyecto de software, existirán mayores posibilidades de corregirlo y menores serán los gastos en el proceso. Resulta fácil entonces, entender la gran importancia de la evaluación arquitectónica para el

buen funcionamiento y la máxima calidad del sistema que se construye. Sin embargo, para que los objetivos de esta sean efectivos, no se puede ignorar la relevancia del momento en que se decida llevar a cabo el proceso.

La evaluación a la arquitectura puede efectuarse en varios momentos dependiendo de su estado, si se encuentra en construcción o ha sido implantada. A dichos momentos se le conocen como evaluación clásica, evaluación temprana y evaluación tardía. La evaluación clásica se efectúa una vez que la arquitectura ha sido terminada y aún no ha sido implementada. La evaluación temprana es llevada a cabo una o varias veces durante la etapa de construcción de la arquitectura, es decir, abarca desde las fases tempranas de diseño y a lo largo del desarrollo del software. Por su parte la evaluación tardía se realiza una vez que la arquitectura existe y la implantación se ha completado. Su principal utilidad está en saber si el sistema cumple con las normas esperadas. (Gómez, 1997)

Clements, Kazman y Klein proponen dos reglas de oro para determinar el momento de efectuar la evaluación, estas son:

1. Realizar una evaluación cuando el equipo de desarrollo inicia la toma de decisiones que afectan directamente a la arquitectura.
2. Cuando el costo de no tomar estas decisiones podría pesar más que el costo de realizar una evaluación. (Carrascoso, 2009)

Determinar la fase en que se realice la evaluación dependerá del responsable de la realización de la misma, siempre teniendo en cuenta que en cada momento se persigue un objetivo que estará dado según la estructura y condiciones del proyecto. Debido a que la investigación se centra en la evaluación de la arquitectura de Cedrux, y esta, a pesar de encontrarse en una etapa muy avanzada en su implementación, aún no se encuentra en un estado terminal. La evaluación en consecuencia se considera temprana. Y está entre las recomendables, ya que es posible en esta etapa la corrección de errores que podrían ser fatales en el futuro.

1.5 Técnicas de evaluación

Las técnicas de evaluación de la arquitectura, son instrumentos o herramientas que conjugadas con otros elementos, proveen a los involucrados en el proceso, de una vía para alcanzar sus objetivos. Estas permiten al mismo tiempo pronosticar el comportamiento de la arquitectura de software en su etapa de diseño. (Milán, 2011)

Un gran grupo de estas técnicas están divididas en cualitativas y cuantitativas. Por lo regular, las técnicas de evaluación cualitativas son utilizadas cuando la arquitectura

está en construcción. Se aplican para la comparación entre arquitecturas candidatas, y tiene relación con la intención de saber la opción que se adapta mejor a cierto atributo de calidad. Este tipo de medición brinda respuestas afirmativas o negativas, sin mayor nivel de detalle.

Las técnicas de evaluación cuantitativa se usan cuando la arquitectura ya ha sido implantada. Estas buscan la obtención de valores que permitan tomar decisiones en cuanto a los atributos de calidad de una arquitectura de software. El esquema general, es la comparación con márgenes establecidos, como lo es el caso de los requisitos de desempeño, para establecer el grado de cumplimiento de una arquitectura candidata, o tomar decisiones sobre ella. Este enfoque permite establecer comparaciones, pero se ve limitado en tanto no se conozcan los valores teóricos máximos y mínimos de las mediciones con las que se realiza la comparación. (Stivens, 2009). En las técnicas cualitativas están comprendidas las técnicas de escenarios, cuestionarios y listas de verificación, mientras que las últimas abarcan las técnicas basadas en métricas, simulaciones, prototipos, experimentos o modelos matemáticos. La figura 2 muestra las diferentes técnicas de evaluación.

Actualmente entre las técnicas de evaluación más utilizadas en el mundo se encuentran las evaluaciones por escenarios, las simulaciones y los modelos matemáticos. (Carrascoso, 2009).

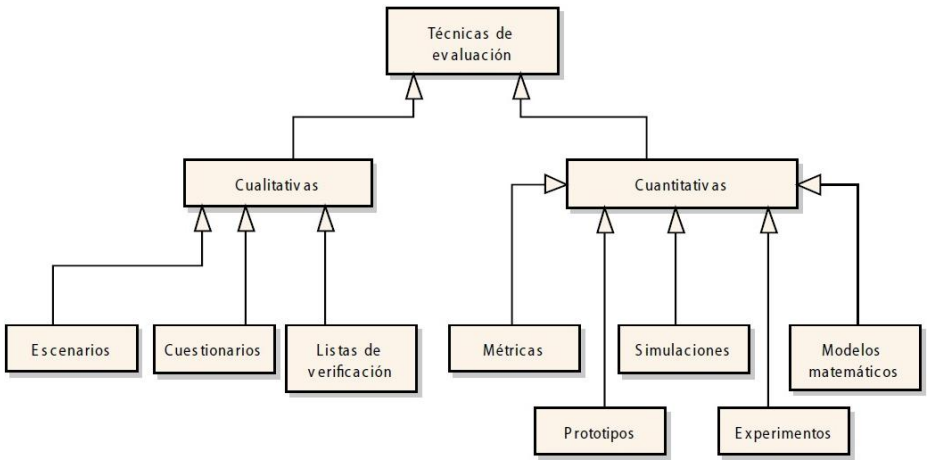

Figura 2. Técnicas de evaluación (Gómez, 1997)

Evaluación basada en escenarios

Un escenario es una breve descripción de la interacción de alguno de los involucrados en el desarrollo del sistema con este. Consta de tres partes:

1. El estímulo: Es la parte del escenario que explica o describe lo que el involucrado en el desarrollo hace para iniciar la interacción con el sistema. Puede incluir ejecución de tareas, cambios en el sistema, ejecución de pruebas, y configuración.
2. El contexto: Describe qué sucede en el sistema al momento del estímulo
3. La respuesta: Describe, a través de la arquitectura, cómo debería responder el sistema ante el estímulo. Este último elemento es el que permite establecer el atributo de calidad asociado. (Kazman, 2001)

Los escenarios proveen un vehículo que permite concretar y entender atributos de calidad. Actualmente las técnicas basadas en escenarios cuentan con dos instrumentos de evaluación relevantes, a saber el Utility Tree y los Profiles. (Camacho, 2004)

Evaluación basada en simulación

La evaluación basada en simulación utiliza una implementación de alto nivel de la arquitectura de software. El enfoque básico consiste en la implementación de componentes de la arquitectura y la implementación a cierto nivel de abstracción del contexto del sistema donde se supone va a ejecutarse. La finalidad es evaluar el comportamiento de la arquitectura bajo diversas circunstancias. Una vez disponibles estas implementaciones, pueden usarse los perfiles respectivos para evaluar los atributos de calidad.

El proceso de evaluación basado en simulación sigue los siguientes pasos:

- ✓ Definición e implementación del contexto.
- ✓ Implementación de los componentes arquitectónicos.
- ✓ Implementación del perfil.
- ✓ Simulación del sistema e inicio del perfil.
- ✓ Predicción de atributos de calidad.

1.5.1 Evaluación basada en modelos matemáticos

La evaluación basada en modelos matemáticos se utiliza para evaluar atributos de calidad operacionales. Permite una evaluación estática de los modelos de diseño arquitectónico, y se presentan como alternativa a la simulación, dado que evalúan el mismo tipo de atributos. Ambos enfoques pueden ser combinados, utilizando los resultados de uno como entrada para el otro. (Erika Camacho, ABRIL – 2004) El proceso de evaluación basado en modelos matemáticos sigue los siguientes pasos:

- ✓ **Selección y adaptación del modelo matemático**. La mayoría de los centros de investigación orientados a atributos de calidad, han desarrollado modelos matemáticos para medir sus atributos. Estos tienden a ser muy elaborados y detallados, así como también requieren de cierto tipo de datos y análisis. Parte de estos datos requeridos no están disponibles a nivel de arquitectura, y la técnica requiere mucho esfuerzo para la evaluación arquitectónica, por lo que el arquitecto de software se ve obligado a adaptar el modelo.
- ✓ **Representación de la arquitectura en términos del modelo**. El modelo matemático seleccionado y adaptado no asume necesariamente que el sistema que intenta modelar consiste de componentes y conexiones. Por lo tanto, la arquitectura necesita ser representada en términos del modelo.
- ✓ **Estimación de los datos de entrada requeridos**. El modelo matemático aun cuando ha sido adaptado, requiere datos de entrada que no están incluidos en la definición básica de la arquitectura. Es necesario estimar y deducir estos datos de la especificación de requisitos y de la arquitectura diseñada.
- ✓ **Predicción de atributos de calidad**. Una vez que la arquitectura es expresada en términos del modelo y se encuentran disponibles todos los datos de entrada requeridos, el arquitecto está en capacidad de calcular la predicción resultante del atributo de calidad evaluado.

Tomando como referencia que en el centro se está realizando una investigación para la aplicación de una técnica de simulación, y por las ventajas que ofrecen los modelos matemáticos, se definió la utilización de esta técnica cuantitativa para la evaluación de la confiabilidad de Cedrux.

Durante el desarrollo de la investigación se analizaron dos modelos matemáticos que están determinados para la evaluación de la confiabilidad. Aunque la investigación no descarta la existencia de otros modelos, no se tiene conocimiento de ellos. Es importante señalar que la principal desventaja de esta técnica de evaluación, es precisamente la inexistencia de modelos matemáticos para la evaluación de los principales atributos de calidad. Esto constituye la causa fundamental de la poca utilización de esta técnica a pesar de sus importantes ventajas.

Modelo basado en redes bayesianas para la evaluación de la confiabilidad de la arquitectura de software.

La línea de productos de software es un enfoque importante en el desarrollo de sistemas de software rentables. Este modelo basado en red bayesiana se presenta para evaluar la confiabilidad de la arquitectura en la línea de montaje. Define la

confiabilidad como la probabilidad de que el sistema pueda realizar sus tareas con algunas restricciones arquitectónicas específicas. El modelo bayesiano se utiliza para evaluar la confiabilidad de cada elemento arquitectónico en tiempo del diseño arquitectónico. El modelo tiene en cuenta los elementos, pero también algunas relaciones en la evaluación de la confiabilidad. Esto resulta muy útil en entornos dinámicos, y sistemas de trabajo en el que las relaciones juegan un papel importante en la confiabilidad de los sistemas. En la figura 3 se ofrece una visión general del modelo.

En las entradas del presente método (los modelos arquitectónicos), se incluyen todas las adaptaciones que figuran el cambio de la concepción, el estado máquina gráfica de los elementos y relaciones, y también la confiabilidad primaria de los artículos y las relaciones. La cantidad de confiabilidad debe estar disponible de antemano como la confiabilidad de los elementos, o puede ser que haya sido obtenida a través de otros métodos disponibles.

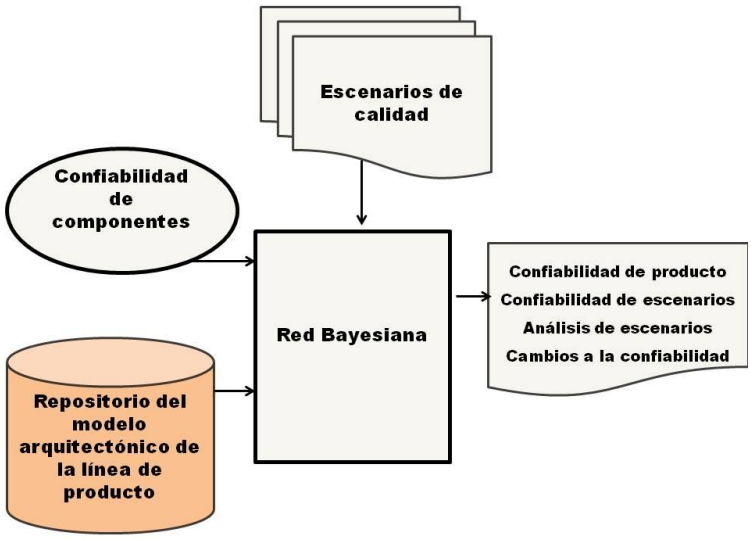

Figura 3. Visión general del modelo (Vaziri, 2011)

Las salidas del método son los siguientes:

- ✓ La confiabilidad de cada conjunto de cambios.
- ✓ La confiabilidad de los productos.
- ✓ La confiabilidad de los escenarios de calidad extraídos.
- ✓ La posibilidad del análisis de los escenarios.

Este modelo plantea que la confiabilidad de las configuraciones de elementos es igual a la probabilidad total de entrada a un estado de fallo. Los pasos generales de este modelo son los siguientes:

En primer lugar, un conjunto de cambios de base (un conjunto de cambios que es común entre todos los productos) se elige y se llevan a cabo los siguientes pasos:

- ✓ Construcción (actualización) de la red bayesiana.
- ✓ La pantalla numérica de red bayesiana: la asignación probable entre los nodos.
- ✓ Obtención de la probabilidad de falla de los nodos.

Usando el gráfico de radar para obtener el efecto acumulativo de las fallas de los nodos disponibles en el escenario de camino. El siguiente paso es la repetición de los pasos mencionados anteriormente para los conjuntos de cambios fundamentales y sus combinaciones. En la combinación de los conjuntos de cambios, solo la red bayesiana se actualiza. (Vaziri, 2011)

Modelo de estimación de la confiabilidad de los componentes de software mediante el aprovechamiento de los modelos arquitectónicos.

Este modelo planteado por (Roshandel, 2006) divide el modelado de un componente en cuatro perspectivas funcionales:

1. La interfaz: Muestra sus servicios prestados y los que se requieren.
2. Comportamiento estático: Muestra la funcionalidad del componente discreto (es decir, en diferentes "Instantes" durante la ejecución del sistema).
3. Comportamiento dinámico: Visión continúa de los detalles de ejecución internos del componente.
4. Interacción del protocolo: Muestra una vista externa continua de un componente en ejecución, especificando la ejecución, permite las huellas de sus operaciones (accede a través de las interfaces).

Lo anterior es utilizado por este modelo de dos maneras importantes. En primer lugar, se utiliza el modelo de comportamiento dinámico como base para el marco de estimación en ausencia de un perfil operacional del componente. En segundo lugar, utiliza el nivel de Interfaz, para mediante la inconsistencia entre los diferentes modelos arquitectónicos, obtener los errores arquitectónicos (o defectos). Estos defectos se utilizan para modelar el comportamiento de fallo del componente.

Plantea además la construcción de un modelo de Markov de interacción entre los estados individuales de un componente, con el fin de estimar la confiabilidad del mismo. Sin embargo, la falta de un perfil de funcionamiento, hace resultar parámetros desconocidos en el mencionado modelo de Markov. Se tiene por consiguiente, que un

Modelo Oculto de Markov (HMM[7]), es un formalismo que puede estimar parámetros ocultos o desconocidos en un modelo, y por lo tanto, este se adecua perfectamente cuando se está en presencia de la falta de un perfil operacional. El marco de estimación de la confiabilidad consta de tres fases (figura 4).

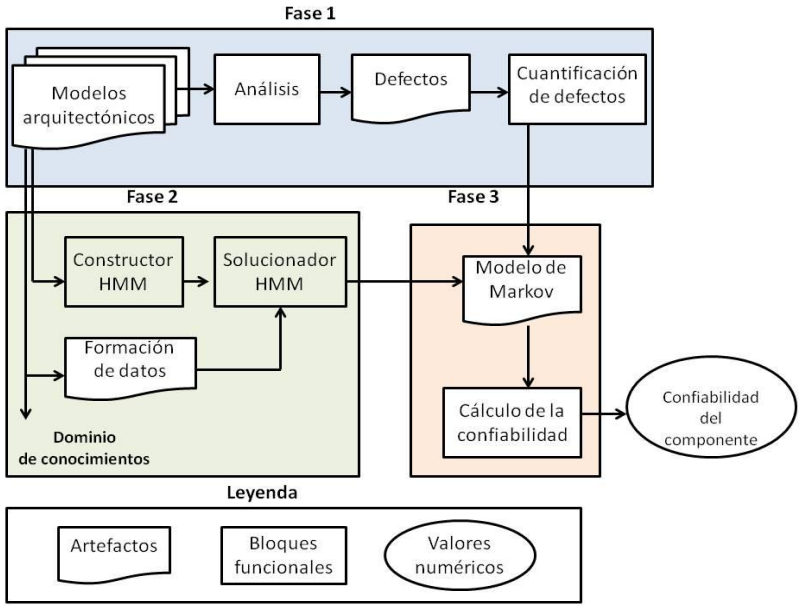

Figura 4. Fases del marco de estimación de la confiabilidad. (Roshandel, 2006)

Fase 1: Modelado de arquitectura, análisis y cuantificación.

En esta fase, se aplican técnicas de análisis estándar del nivel de los modelos de arquitectura, y se identifican los defectos arquitectónicos del componente. Del enfoque arquitectónico multi-vista a componente resulta el modelo de Interfaz con las inconsistencias en las vistas arquitectónicas. Este modelo se aprovecha para representar a los defectos. Los defectos se basan en la clasificación que se muestra en la figura 5, y para su cuantificación el modelo propone un marco de costo. La relación marco -costo utiliza un costo funcional para cuantificar los defectos, sobre la base de las observaciones de que defectos diferentes afectan la confiabilidad de un componente de diferentes maneras, y que los costos relacionados con la mitigación del defecto varían según el tipo de defecto.

[7] Hidden Markov Model.

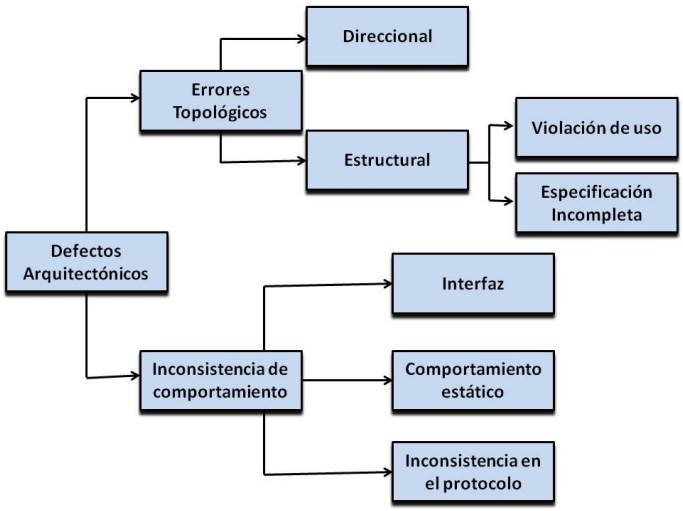

Figura 5. Clasificación de errores (Roshandel, 2006)

En su nivel superior, los defectos arquitectónicos se dividen en dos clasificaciones:

- ✓ Errores topológicos.
- ✓ Inconsistencias de comportamiento.

Errores topológicos: Tienden a ser globales en la arquitectura y se refieren a aspectos relacionados con la configuración de componentes y conectores en el sistema. A menudo, son resultados de la violación de las restricciones impuestas por los estilos arquitectónicos. Los errores topológicos son de naturaleza direccional o estructural:

La dirección: Específica si la comunicación necesaria por el estilo arquitectónico es violada. Un ejemplo de este error es cuando en una arquitectura cliente-servidor el servidor solicita un servicio a partir de un cliente.

Estructura: En su naturaleza se dividen en Violaciones de uso y Especificación incompleta.

- ✓ Violación de uso: Un ejemplo de una violación de uso es cuando un enlace de comunicación entre dos componentes está ausente, o, alternativamente, cuando una comunicación o relación entre los componentes existe donde no debe estar presente.
- ✓ Especificación incompleta: Se manifiesta cuando hay principalmente información insuficiente para especificar las propiedades de los componentes del modelo arquitectónico y conectores. (errores internos en los componentes por pobre delimitación de sus responsabilidades)

Inconsistencias de comportamiento: Se relacionan con la información arquitectónica local a un componente, un servicio de un conector, o un conjunto de componentes que interactúan. Se refieren a los desajustes entre las interfaces, los comportamientos estáticos, o protocolo de interacción de componentes. Por su naturaleza se dividen en:

- ✓ Interfaz: Un defecto de interfaz ocurre cuando las etiquetas de servicios proporcionados y/o requeridos entre dos componentes no coinciden.
- ✓ Inconsistencia estática del comportamiento: Revela desajustes entre las pre y post-condiciones proporcionadas correspondientemente y servicios que se requieren en dos componentes.
- ✓ Inconsistencia en el protocolo: Pone de manifiesto los errores de interacción de protocolos entre los componentes.

Los valores obtenidos a partir de la cuantificación de defectos, se utilizan en la fase de cálculo componente de confiabilidad.

Fase 2: Modelado del perfil Operacional.

El desconocido perfil operacional del componente durante el diseño arquitectónico induce incertidumbre en el proceso de estimación de la confiabilidad. Este enfoque de modelado de confiabilidad por lo tanto incluye una etapa en la que, dada la información disponible sobre el componente, se construye un HMM que aprovecha directamente el modelo del comportamiento dinámico del componente. En particular, se mapea el modelo de comportamiento dinámico del componente a un HMM de la siguiente manera: los estados en el modelo de comportamiento dinámico convertido en los estados correspondientes en el HMM (conjunto S), y el evento pares de acción del modelo de comportamiento dinámico convertido en observaciones del HMM (juego O). El solucionador de HMM después aprovecha un conjunto de sintetizados o simulados datos de entrenamiento y se aplica el conocido algoritmo de Baum Welch para estimar los parámetros desconocidos. Estos parámetros son probabilidades de transición, que conceptualmente corresponden al perfil de operación del componente. Es importante señalar aquí que la cuantificación del defecto y la formación y generación de datos son a la vez módulos enchufables del marco. En otras palabras, el método utilizado para cualquiera de estos procesos es independiente de la propia estructura.

Fase 3: Cálculo de la confiabilidad.

Al resolver el HMM y la estimación de los parámetros desconocidos, es un modelo de Markov obtenido. Este modelo se extiende entonces con un estado de fallo F que representa el estado del componente cuando se produce un error. Se añade una transición de cada estado en el modelo para el estado de fallo, y designa un comportamiento erróneo.

Las probabilidades de las transiciones para el estado de fallo se obtienen a partir de la etapa de cuantificación de defectos. También se añade una sola transición desde el estado de la falla a un designado estado en el modelo, para representar a la recuperación de errores. (Roshandel, 2006)

1.5.2 Consideraciones generales de los modelos de evaluación descritos

Tabla 3. Comparación de modelos matemáticos para evaluar la confiabilidad de un sistema.

Indicadores	Modelo basado en redes bayesianas	Modelo de estimación mediante matriz de Markov
Evaluación a nivel de sistema.	**X**	
Entrada de datos de entrenamiento		**X**
Flexibilidad de modificación de métodos de cálculo.		**X**
Necesidad de evaluación previa componentes de la AS	**X**	**X**
Basado en fallos o errores del sistema.		**X**

A pesar de que los modelos matemáticos comparados anteriormente tienen la ventaja en común que se centran en la evaluación cuantitativa de la confiabilidad, se ha llevado a cabo una comparación entre ellos (tabla 3).

En esta tabla se observa que el modelo basado en redes bayesianas realiza su evaluación a nivel de sistema. Lo anterior hace necesario que antes de la utilización de este, se cuente con datos de análisis de confiabilidad de los elementos. Los datos pueden ser obtenidos a través de otros métodos de evaluación. Esta característica se convierte en una desventaja frente al modelo de estimación mediante la matriz de Markov, que evalúa la confiabilidad de un componente. Es decir, este último modelo parte de la base sin necesidad de evaluación previa, que es lo que se requiere

teniendo en cuenta que no existen evaluaciones cuantitativas a la confiabilidad de Cedrux. Aunque el segundo modelo presenta la característica que necesita una entrada de datos de entrenamiento, para la construcción del Modelo Oculto de Markov, tiene a su vez la ventaja de que es flexible a modificaciones presentando módulos enchufables a la hora de definir los cálculos, lo que solventa la desventaja anterior. Está basado en el análisis de errores o fallos que son tomados como entrada, el modelo de estimación mediante la matriz de Markov permite determinar los errores de mayor peso que afectan el sistema, de modo que brinda la visión del error en aras de facilitar el proceso de corrección contribuyendo a agilizar el proceso de desarrollo del sistema.

Después de lo analizado anteriormente, por sus visibles ventajas se determinó tomar el Modelo de estimación de la confiabilidad de los componentes de software mediante el aprovechamiento de los modelos arquitectónicos, como guía para esta investigación.

1.6 Conclusiones parciales

Con la realización de este capítulo se arribaron a las siguientes conclusiones:

- ✓ La confiabilidad es un atributo de calidad que desempeña un papel fundamental en los sistemas de gestión integral. A medida que estos sistemas crecen y se hacen complejos, crecen también las probabilidades de fallos, de ahí la importancia de una evaluación exitosa de la confiabilidad.
- ✓ Los modelos de calidad son elementos esenciales en el proceso de evaluación, el modelo que guía la investigación es el modelo ISO/IEC 9126. Este modelo cuenta con un alto prestigio a nivel internacional, además de ser el definido por la Oficina Nacional de Normalización para la producción de sistemas de software.
- ✓ La evaluación de la confiabilidad mediante modelos matemáticos puede brindar una visión del estado de este atributo en la arquitectura y a su vez retribuir el proceso de desarrollo del software.
- ✓ El modelo de evaluación de la confiabilidad de un componente del sistema Cedrux, se definirá a partir del Modelo de estimación de la confiabilidad de los componentes de software mediante el aprovechamiento de los modelos arquitectónicos**.**

Capítulo II: Propuesta de Solución

El presente capítulo se centrará en reconocer y estudiar el modelo matemático identificado en el primer capítulo, con el fin de describir las prácticas que se adecuen al sistema, así como las características que no se adapten a este. En consecuencia, durante el desarrollo del capítulo se detallan las modificaciones necesarias en cada fase del modelo, en busca de la realización de un proceso de evaluación de la confiabilidad preciso y concreto. En este capítulo además, se mencionan y ejemplifican las ventajas que brinda el modelo a los arquitectos del sistema en aras de agilizar el proceso de desarrollo del software.

2.1 Análisis de las bases del modelo respecto al desarrollo de Cedrux

El modelo matemático propuesto, como base de estudio para esta investigación, presenta características y módulos funcionales primordiales para la evaluación de la confiabilidad. Sin embargo, para la utilización de un modelo, es crucial su estudio, en busca de aquellos factores que sean relevantes en el sistema, como también los que no se adaptan a las particularidades de lo que se quiere evaluar. Resulta entonces importante la realización de un análisis de este modelo, con el fin de determinar lo que es conveniente y lo que no se adapta al sistema Cedrux.

Buenas prácticas del Modelo de estimación de la confiabilidad de los componentes de software mediante el aprovechamiento de los modelos arquitectónicos.

1. La evaluación a nivel de componente es ventajosa, teniendo en cuenta que Cedrux sigue una filosofía de arquitectura orientada a componente, encaminada a lograr una línea de productos de software.
2. La división del modelo en tres fases, y la estructuración de estas sirviendo las primeras dos fases como entrada a la tercera, permite la organización del modelo en módulos funcionales que son fácilmente modificados a conveniencia del modelo a definir.
3. El análisis y detección de errores llevado a cabo en la primera fase, tiene gran relevancia para el equipo de desarrollo, una vez identificados los errores resulta viable la corrección de estos.

4. La asignación de pesos a los errores del sistema, brinda una visión de la importancia de los mismos, es decir, el impacto de estos sobre el sistema, de manera que se puede fácilmente establecer un orden a seguir para la eliminación de estos fallos
5. El modelo de comportamiento dinámico permite, mediante máquinas de estados, representar el comportamiento del componente luego de haberse producido el error, esto tiene gran significación porque brinda una visión de las consecuencias de ese error para el componente.

Prácticas del Modelo de estimación de la confiabilidad de los componentes de software mediante el aprovechamiento de los modelos arquitectónicos que no se adecúan al sistema Cedrux.

1. La representación por vistas del componente que plantea el modelo, no se corresponde con la clasificación de las vistas presentes actualmente en Cedrux, Se requieren modificaciones para lograr que estas sean homólogas.
2. La clasificación de los defectos o errores arquitectónicos no se ajusta totalmente al sistema.
3. Para la asignación de pesos a los defectos, el modelo plantea un marco de costo basado en experiencias de evaluaciones anteriores a la arquitectura. La investigación no cuenta con este marco, debido a la falta de evaluaciones cuantitativas a la confiabilidad de Cedrux.
4. En la segunda fase del modelo, una vez construido el Modelo Oculto de Markov, el solucionador del HMM utiliza un conjunto de datos de entrenamiento para aplicar el algoritmo de Baum-Welch. Estos datos están basados en evaluaciones previas. Teniendo en cuenta que no existen evaluaciones a la arquitectura de Cedrux mediante modelos matemáticos, que puedan propiciar resultados cuantitativos, se ve comprometida la realización de la segunda fase del modelo.
5. En la tercera fase para construir la matriz de Markov a partir de la cual se van a realizar los cálculos de la confiabilidad, se requieren los datos provenientes del solucionador del HMM, lo que queda imposibilitado por lo descrito en el punto anterior.
6. El modelo tiene además entre sus principales desventajas, que a pesar de brindar resultados cuantitativos que dan la medida del estado de confiabilidad del componente, no es capaz de mostrar interpretaciones viables que asistan el proceso de desarrollo.

Según (Bosch, 2000), para llevar a cabo un proceso de evaluación mediante modelos matemáticos el primer paso consiste en la selección y adaptación del modelo (Ver epígrafe 1.4.2).

Ya seleccionado el Modelo de estimación de la confiabilidad de los componentes de software mediante el aprovechamiento de los modelos arquitectónicos, y además identificadas las principales ventajas y desventajas de este para Cedrux, se requiere de la adaptación del modelo. La misma tendrá como primicia mantener las buenas prácticas, así como adecuar los diferentes factores que no se adaptan al sistema.

2.2 Propuesta de modificación al modelo matemático estudiado

En el epígrafe anterior se determinó la necesidad de adaptar el modelo. Este proceso de adaptación para Cedrux, requiere de una serie de modificaciones que a continuación se dividen y explican según cada una de las fases con las que cuenta el modelo propuesto.

2.2.1 Fase1. Modelado de la arquitectura, análisis y cuantificación

En la primera fase del modelo, como su nombre lo indica, se realiza el modelado de la arquitectura a evaluar. Para el modelo de evaluación se requiere la utilización de dos vistas.

1. Interfaz: Está definida en Cedrux como de Integración.
2. Comportamiento dinámico: Tiene como homóloga en el sistema Cedrux la Vista de sistema.

En esta fase además se lleva a cabo un análisis en busca de la detección y cuantificación de errores que contenga el componente a evaluar. Con el objetivo de facilitar las modificaciones necesarias descritas en el epígrafe 2.1, se realizó una encuesta (Ver anexo 2) a nueve arquitectos de Cedrux. Teniendo en cuenta que estos muestran conocimientos del sistema al que va dirigido el modelo a definir, se decidió tomar como punto de referencia sus criterios, con el fin de adaptar la clasificación de cada uno de los errores en correspondencia con el sistema. En esta encuesta también se establecen los costos de los errores según su clasificación, en dependencia de lo dañino que puede resultar cada error en el sistema. Esta fase se muestra en la figura 6, y en ella se van a realizar las siguientes actividades:

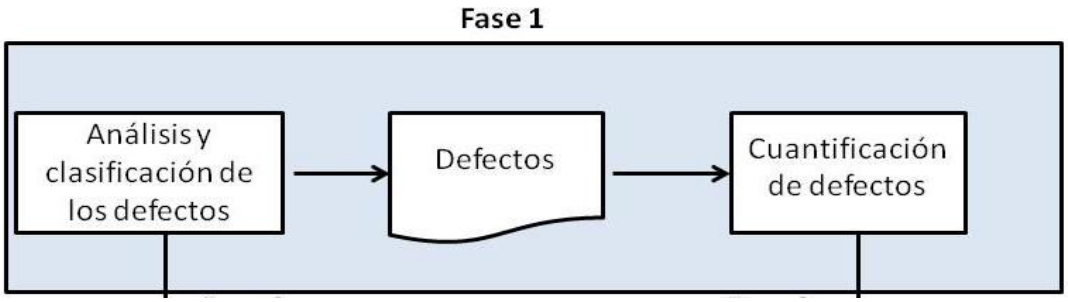

Figura 6. Fase1 del Modelo de estimación de la confiabilidad de los componentes de software mediante el aprovechamiento de los modelos arquitectónicos. (Roshandel, 2006)

1. Análisis y clasificación de los defectos o errores del componente.

El análisis estará encaminado a detectar los errores que estén presentes en el componente. A partir del análisis realizado, y teniendo ya determinados los errores, se va a proceder a clasificar cada uno de ellos. La clasificación de estos defectos para el modelo, luego de las transformaciones realizadas por los arquitectos, quedó definida como se muestra en la figura 7. Clasificado cada uno de los errores según lo ya expuesto, entonces se podrá pasar a la segunda parte de esta fase, la cual inicia con la clasificación para asignar un valor ya determinado a cada uno de ellos.

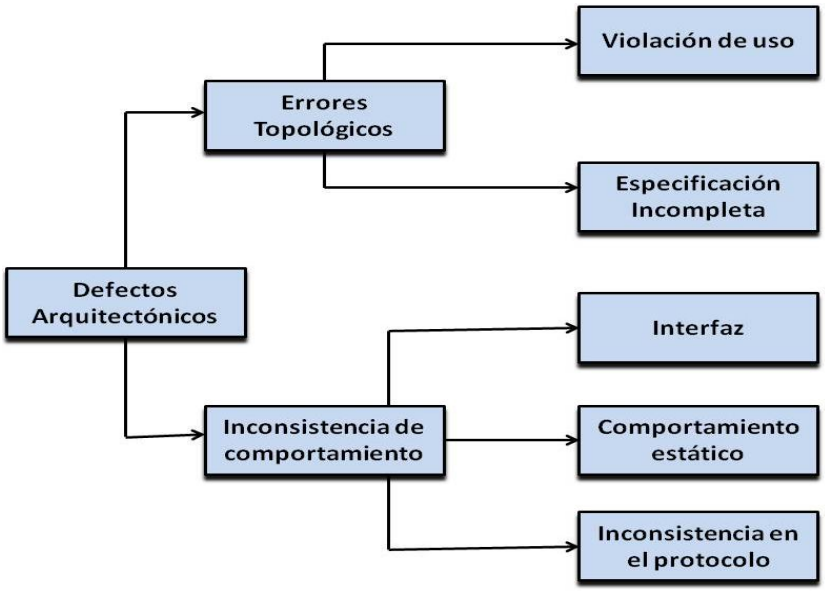

Figura 7. Clasificación de errores determinada para el nuevo modelo definido.

2. Cuantificación de defectos.

Los resultados de la encuesta realizada ofrecen una nueva asignación de pesos para los errores, de manera que mediante esta se puedan identificar los errores que puedan afectar significativamente la arquitectura del sistema. En la tabla 4 se muestra la nueva cuantificación de defectos definida, como resultado de la asignación de costo a cada clasificación. A efectos matemáticos se estableció que el impacto de los costos será

determinado de la siguiente manera: a mayor valor de costo, mayor afectación del valor final de la confiabilidad (impacto negativo sobre la confiabilidad).

Tabla 4. Cuantificación de errores según su clasificación

Error según su clasificación	Costo del error	Descripción
Interfaz	1.0	Ocurre cuando las etiquetas de los servicios proporcionados y/o requeridos entre dos componentes no coinciden.
Inconsistencia estática del comportamiento	0.8	Revela desajustes entre las pre y post-condiciones proporcionadas correspondientemente a servicios que se requieren en dos componentes.
Inconsistencia en el protocolo	0.5	Pone de manifiesto los errores de interacción de protocolos entre los componentes.
Especificación incompleta	0.2	Manifiesta información insuficiente para especificar las propiedades de los componentes del modelo arquitectónico y conectores. Errores internos en los componentes por pobre delimitación de sus responsabilidades.
Violación de uso	0.1	Aparece cuando un enlace de comunicación entre dos componentes está ausente, o, alternativamente, cuando una comunicación o relación entre los componentes existe donde no debe estar presente.

Esta fase tiene como resultado la clasificación y el peso de los defectos identificados en el análisis, y va a servir de entrada a las fases dos y tres respectivamente.

2.2.2 Fase 2. Modelado del perfil operacional

Teniendo en cuenta que la formación y generación de datos son a la vez módulos enchufables, se ha decidido la realización de importantes transformaciones que dividen la segunda fase en un módulo principal (figura 8).

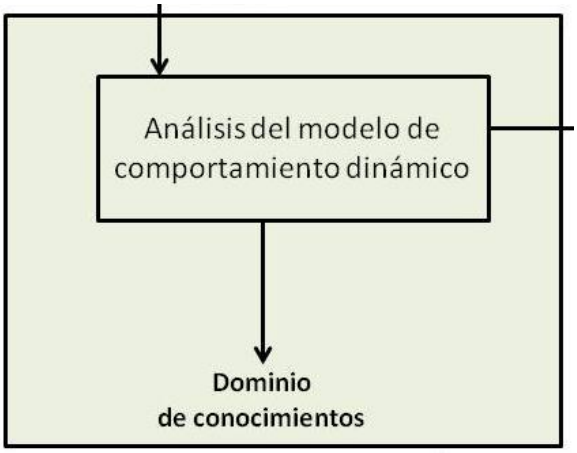

Figura 8. Fase2 del Modelo de estimación de la confiabilidad de los componentes de software mediante el aprovechamiento de los modelos arquitectónicos (Roshandel, 2006)

1. **Análisis del modelo de comportamiento dinámico.**

Este modelo va a estar representado por un grafo dirigido (figura 9). Cada nodo del grafo representa un estado transitorio por el que puede pasar el componente durante un fallo o error (tabla 5). La recuperación y el fallo total son los estados terminales del grafo, y los posibles estados finales del modelo de comportamiento dinámico. Estos estados se definieron a partir de un estudio realizado en el que se consultó (Extremadura, 2008), identificándose de este modo posibles estados que se adaptaron al sistema, a partir del criterio de los arquitectos de Cedrux. El grafo dirigido va a brindar la oportunidad de recorrer un posible camino por el que puede transitar el componente producto al fallo, desde que se produce el error hasta llegar al fallo total o a la recuperación. Cada camino va a tener un peso o costo en dependencia de su impacto en el sistema.

Para cada subcaracterística madurez, tolerancia a fallos, recuperabilidad (ISO/IEC, 1998), se estableció un orden por cada camino posible a recorrer, este orden se muestra en las tablas 6, 7 y 8 respectivamente. Empieza en el camino más factible terminando en el camino que puede ser nefasto para el componente provocando un fallo total

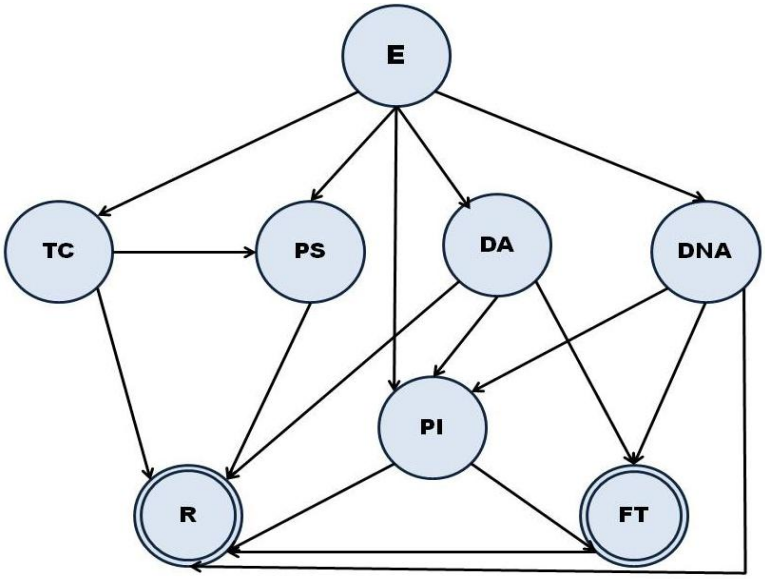

Figura 9. Modelo de comportamiento dinámico para la evaluación de la confiabilidad de la arquitectura de Cedrux.

A partir del orden de estos caminos, el peso de estos variará según la subcaracterística de la confiabilidad a evaluar. Es decir, cada camino, tendrá un costo diferente para cada subcaracterística, teniendo en cuenta el nivel de factibilidad de ese camino para esta. Es importante destacar que para cada subcaracterística el mejor y el peor camino a recorrer pueden ser iguales o diferentes teniendo en cuenta su concepto según la ISO/IEC 9126.

Tabla 5. Estados o nodos del modelo comportamiento dinámico

Estados	Descripción
Tolerancia completa (TC)	El componente sigue funcionando, sin pérdida de funcionalidad.
Degradación Aceptable (DA)	El componente presenta pérdida de funcionalidad, pero continúa con una ejecución aceptable.
Degradación No Aceptable (DNA)	El componente presenta pérdida de funcionalidad, manteniendo un nivel de ejecución no aceptable.
Parada segura (PS)	El componente se detiene de forma segura sin pérdida de funcionalidad.

Parada insegura (PI)	El componente se detiene de forma insegura perdiendo sus funcionalidades.
Fallo total (FT)	El componente colapsa.
Recuperación (R)	El componente se recupera del fallo.

La relevancia de la ponderación de los caminos del grafo para cada subcaracterística, se basa en la teoría de que el peso de estos, va a establecer su importancia. De esta manera, para un error se determina un camino que será la traducción de los posibles estados por los que pueda pasar el componente a partir del error. El peso del camino va a determinar la confiabilidad de la subcaracterística que se evalúa. Establecido en un rango de 0 a 10, los mejores valores de la confiabilidad serán los más cercanos a 0, mientras que los allegados a 10, son los que tendrán mayor impacto negativo sobre el sistema (figura 10).

Tabla 6. Orden de prioridad de los caminos del grafo, y costo de ellos para la Madurez, partiendo del camino más factible

No.	Estado 1	Estado2	Estado3	Estado4	Estado5	Costo
1.	E	TC	R	-		1
2.	E	TC	PS	R		2
3.	E	PS	R			2.5
4.	E	DA	R			3
5.	E	DNA	R			5
6.	E	DA	PI	R		6
7.	E	DNA	PI	R		7.5
8.	E	PI	R			8
9.	E	DA	PI	FT		10
10	E	DA	PI	FT	R	10
11	E	DA	FT			10
12	E	DA	FT	R		10
13	E	PI	FT			10
14	E	PI	FT	R		10
15	E	DNA	FT			10
16	E	DNA	PI	FT		10

17	E	DNA	PI	FT	R	10
18	E	DNA	FT	R		10

Tabla 7. Orden de prioridad de los caminos del grafo, y costo de ellos para la Tolerancia a fallos, partiendo del camino más factible.

No.	Estado 1	Estado2	Estado3	Estado4	Estado5	Costo
1.	E	TC	R	-		1
2.	E	DA	R			2
3.	E	TC	PS	R		2.5
4.	E	PS	R			3
5.	E	DA	PI	R		5
6.	E	PI	R			6
7.	E	DNA	R			7
8.	E	DNA	PI	R		8
9.	E	DA	PI	FT		10
10.	E	DA	PI	FT	R	10
11.	E	DA	FT			10
12.	E	DA	FT	R		10
13.	E	PI	FT			10
14.	E	PI	FT	R		10
15.	E	DNA	FT			10
16.	E	DNA	PI	FT		10
17.	E	DNA	PI	FT	R	10
18.	E	DNA	FT	R		10

Tabla 8. Orden de prioridad de los caminos del grafo, y costo de ellos para la Recuperabilidad partiendo del camino más factible.

No.	Estado 1	Estado2	Estado3	Estado4	Estado5	Costo
1.	E	TC	R	-		1
2.	E	DA	R			2
3.	E	PS	R			2.5
4.	E	TC	PS	R		3
5.	E	DNA	R			5
6.	E	PI	R			5.5
7.	E	DA	PI	R		5.8
8.	E	DNA	PI	R		6.5

9.	E	DA	FT	R		8
10	E	DNA	FT	R		8.1
11	E	PI	FT	R		8.2
12	E	DA	PI	FT	R	8.3
13	E	DNA	PI	FT	R	8.5
14	E	PI	FT			10
15	E	DNA	FT			10
16	E	DNA	PI	FT		10
17	E	DA	PI	FT		10
18	E	DA	FT			10

Es válido aclarar, la posibilidad de ocurrencia de que un error pueda conducir a varios caminos en el modelo de comportamiento dinámico. Sin embargo, el modelo solo va a evaluar un camino. Independientemente, el arquitecto puede medir la confiabilidad para ambos caminos, de modo que se pueda establecer comparaciones entre la confiabilidad de los caminos determinados.

Los caminos además van a estar divididos en tres rangos. La tabla 9 esclarece lo anterior.

Tabla 9. Rangos de confiabilidad para los caminos del grafo

Confiabilidad	Rango	Descripción
Caminos Confiables	0-3.	Caminos que no provocan un nivel de ejecución no aceptable o colapsos del sistema.
Poco Confiables	5-8	Caminos que a pesar de siempre llegar a la recuperación pierden funcionalidades
No confiables	10	Caminos que llegan al fallo total haciendo colapsar el sistema

El procedimiento antes descrito se va a realizar para cada error que se detecte en la primera fase del modelo. La segunda fase tendrá como salida todos los errores cuantificados y la confiabilidad para las tres subcaracterísticas, dada por el peso del camino para cada una de ellas.

2.2.3 Fase3. Cálculo de la confiabilidad

La tercera fase tiene como salida la confiabilidad del componente (figura 10). El modelo definido tiene como entrada la cuantificación de los errores de la primera fase, y la confiabilidad de cada subcaracterística de la segunda fase. En esta fase se requiere la realización de cálculos, que permitan un análisis de la confiabilidad, de modo en que a partir de este, se puedan identificar los errores más costosos al sistema, y adoptar decisiones en consecuencia que los mitiguen. La tercera fase del modelo quedaría reestructurada de la siguiente manera (Ver figura 10):

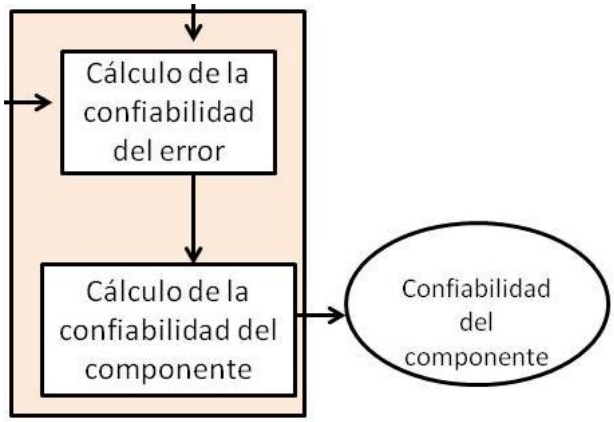

Figura 10. Fase3 del Modelo de estimación de la confiabilidad de los componentes de software mediante el aprovechamiento de los modelos arquitectónicos (Roshandel, 2006)

1. **Cálculo de la Confiabilidad del componente para un error.**

Obtenida la cuantificación de los errores determinados en la primera fase, y además de la segunda fase el resultado de la confiabilidad para cada subcaracterística a partir del camino recorrido en el modelo de comportamiento dinámico; se estará en condiciones de aplicar o realizar los cálculos que darán paso a los valores cuantitativos a partir de la fórmula que se muestra a continuación:

$$C_{CE} = \frac{C_M + C_{TF} + C_R}{3} \times P_E$$

En la tabla 10 se muestran las variables de la fórmula descrita.

Tabla 10. Variables para el cálculo de la confiabilidad del componente ante un error

Variables	Descripción	
P_E	Peso del error	Determinado según la clasificación a la que pertenezca el error.

C_M	Confiabilidad de la madurez	Determinada por el peso del camino del grafo para la Madurez.
C_{TF}	Confiabilidad de la tolerancia a Fallos	Determinada por el peso del camino del grafo para la Tolerancia a fallos
C_R	Confiabilidad de la recuperabilidad	Determinada por el peso del camino del grafo para la Recuperabilidad.
C_{CE}	Confiabilidad del componente ante el error o fallo	Establecida a partir de la fórmula que relaciona las variables anteriores.

Este procedimiento va a aplicarse a cada error detectado, de modo que para cada error se va a obtener un valor de la confiabilidad del componente a partir del mismo. Esta confiabilidad va a encontrarse en un rango de 0 hasta 10. Partiendo de la filosofía siguiente:

- ✓ Confiables: Son aquellos errores en los que el componente se recupera sin llegar a un paro, o a un estado de ejecución no aceptable.
- ✓ Poco confiables: Errores que pierden funcionalidades, o paran de forma insegura, pero nunca llegan al fallo total.
- ✓ No confiables: Son los llegan al fallo total provocando el colapso del sistema.

Luego de un estudio de las confiabilidades posibles para cada uno de estos errores, se determinó un rango en el que oscilan las confiabilidades de cada uno de ellos. Este se muestra en la escala de la figura 11. Esta escala servirá como punto de comparación para identificar los errores de mayor riesgo para el sistema, teniendo en cuenta que a mayor resultado, mayor será la afectación en el componente. Esta deducción va a ser de gran ayuda para el equipo de desarrollo.

Figura 11. Escala para la evaluación del valor de la confiabilidad para los errores detectados en el componente.

2. Interpretación de la confiabilidad del componente.

En el paso anterior se obtiene el cálculo de la confiabilidad del componente para cada error. Sin embargo, este paso no será suficiente para brindar una interpretación del estado de la confiabilidad del componente.

El modelo definido, plantea la utilización del algoritmo de agrupamiento de las K-medias. Básicamente este algoritmo busca formar clusters[8] los cuales serán representados por K objetos. Cada uno de estos K objetos son el valor medio de los objetos que pertenecen a dicho grupo. A continuación se enumeran los pasos para llevar a cabo este algoritmo. (Fernández, 2010)

1. Inicialmente se seleccionan K objetos del conjunto de entrada. Estos K objetos serán los centroides iniciales de los K-grupos.
2. Se calculan las distancias de los objetos (datos) a cada uno de los centroides. Los datos (objetos) se asignan a aquellos grupos cuya distancia es mínima con respecto a todos los centroides.
3. Se actualizan los centroides como el valor medio de todos los objetos asignados a ese grupo.
4. Se repite el paso 2 y 3 hasta que se satisface algún criterio de convergencia.

Este algoritmo, se aplicará en el modelo luego de tener la lista de confiabilidades determinadas a los errores detectados. En cada uno de los siguientes puntos se describe ejecución de este en el modelo.

1. Toma como conjunto inicial las confiabilidades obtenidas a partir de los errores detectados en el componente. Inicialmente cada elemento (confiabilidad), del conjunto de entrada, va a representar un grupo y cada dato será el centroide de su propio grupo.
2. Se calculan las distancias entre las confiabilidades obtenidas, y se agrupan según su cercanía respecto a los centroides.
3. Se busca el valor medio de cada grupo de confiabilidades obtenido en el paso anterior y se actualiza como los centroides de cada grupo (3 grupos). De este

[8] Grupos usados para representar un conjunto de valores, entre todos los iniciales que tiene algo en común, y se pueden agrupar en función de determinado rango (Fernández, 2010).

modo se podrán conocer los valores que se encuentran contenidos en cada unos de ellos (altos, medios o bajos).

4. Se cuenta cada elemento de cada conjunto para saber la cantidad de confiabilidades pertenecientes a cada grupo, con el fin de conocer la cantidad de valores confiables y los que por el contrario pueden ser de gran impacto para el sistema (Los máximos valores).
5. En caso de que no sea posible la división de los datos en 3 grupos entonces será necesario disminuir el número de grupos. El procedimiento restante será el mismo.

Luego de la obtención de estos elementos la confiabilidad del componente estará dada por fórmula que se muestra a continuación, y que representa la suma de los centroides obtenidos. La tabla 11 describe las variables presentes en la fórmula para el cálculo de la confiabilidad del componente.

$$Cc = \sum_{i=0}^{i=n} \text{CE}$$

Tabla 11. Variables para el cálculo de la confiabilidad del componente.

Variables	Descripción
Cc	Confiabilidad del componente
CE	Centroide

En fases anteriores se puntualizó que en el caso del error así como los caminos del modelo dinámico, a menor costo, menor será el impacto sobre el sistema, por lo que la confiabilidad de componente será tan confiable como cercano a 0 sea su valor y viceversa. El valor final deberá ser evaluado en la escala mostrada en la tabla 12.

Partiendo del conocimiento de todos los posibles centroides que puede devolver el modelo se puede definir el rango en el que el componente va a ser confiable, poco confiable, o no confiable. Este rango va a fluctuar entre 0 y el máximo valor que puede alcanzar la confiabilidad de un componente en el modelo.

Tabla 12. Especificación de los rangos de evaluación de las confiabilidades de los componentes.

Rangos	Confiabilidad del componente	Descripción
0 – 1.30	Confiables	Posible mayor centroide (1.30). Incluye las confiabilidades de los errores que llegan a la recuperabilidad con un nivel de ejecución aceptable.
1.31 - 7.0	Poco Confiables	La suma del mayor centroide de los confiables (1.30), y el mayor centroide posible de los errores poco confiables (5.70). Incluye todos los errores que no llegan nunca al fallo total.
7.01- 17.01	No Confiables	La suma de los dos mayores centroides anteriores, (7.0) y el mayor centroide de los no confiables (10). Incluye los caminos que llegan al fallo total y son nefastos para el sistema.

La figura 12 muestra la estructura del modelo luego de las modificaciones realizadas. En esta figura se divisan las tres fases del modelo y los módulos a realizar en cada una de ellas, hasta llegar a la fase final a partir de la cual se obtiene el valor de la confiabilidad del componente. Queda de este modo constituido el modelo definido por la investigación.

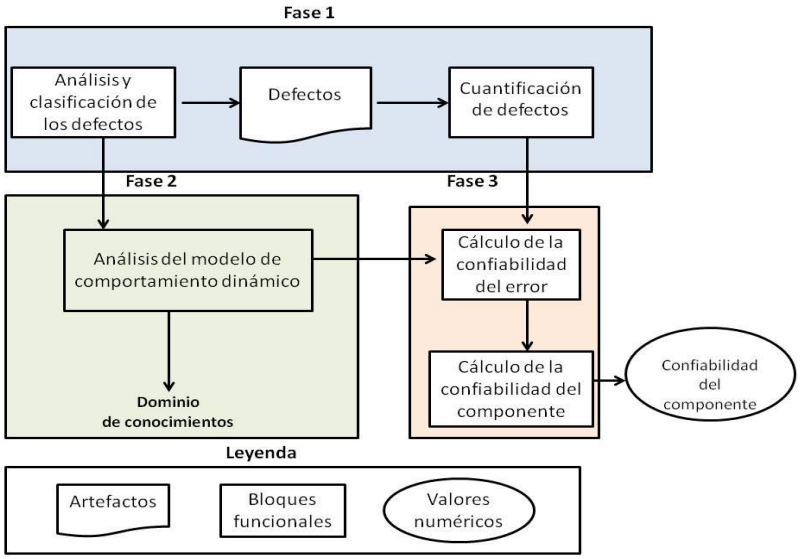

Figura 12. Modelo propuesto para evaluar la confiabilidad de un componente de la arquitectura de Cedrux.

2.3 Restricciones del modelo definido por la investigación

1. Cada tipo de error incluido en la investigación, deberá contar con un peso por el que se pueda establecer la prioridad de este entre el resto, dentro de una clasificación determinada.
2. La cuantificación de cada error debe estar acorde con el tipo de error que se esté analizando, siendo consecuente con que a mayor valor, peores serán las consecuencias de ese error al componente.
3. Los posibles caminos por subcaracterísticas se ordenan a partir del camino óptimo y hasta el menos factible, estableciendo costos menores a los más factibles y viceversa.
4. Cada camino definido presenta una ponderación diferente para cada subcaracterística.
5. Los caminos tendrán un rango de valores desde 0 hasta 10, y los errores entre 0 y 1.
6. El arquitecto evaluador no podrá incluir caminos al grafo fuera de los que ya están definidos.

2.4 Relevancia del modelo definido para el proceso de desarrollo

El modelo definido centra su principal ventaja en brindar resultados que puedan retribuir el proceso de desarrollo. Durante el desarrollo de cada una de las fases por las que se transita para llegar al cálculo final de la confiabilidad, se arriban a conclusiones que son indudablemente beneficiosas para la toma de decisiones del equipo de desarrollo. A continuación se describen y ejemplifican cada una de ellas según la fase a la que corresponda.

La primera fase de este modelo, tiene como salida todos los errores que se puedan detectar en el componente, y el costo que se le asigne según la clasificación a la que pertenezcan. El ejemplo de la tabla 13 muestra un análisis a un componente que tuvo como salida tres errores de clasificaciones diferentes.

Tabla 13. Errores clasificados y con peso asignado (ejemplo).

No.	Errores detectados y clasificado	Descripción del error	Peso asignado

1.	Error de Violación de uso	La vista en Capital Humano invoca a algún modelo sin pasar por el controlador	0.1
2.	Especificación incompleta.	Incorrecto uso de la Herencia. Existen clases hijas que reproducen funciones de la clase padre.	0.2
3.	Error de Interfaz	Capital Humano requiere servicio Y de Inventario, cuando debería ser servicio X.	1

Mediante el conocimiento de los errores, así como el valor asignado a cada uno de ellos, el arquitecto puede arribar a importantes interpretaciones mencionadas a continuación:

- ✓ Determinar el tipo de error según la clasificación, que está incidiendo mayormente en el componente.
- ✓ Medir la influencia de cada error según el peso establecido a su clasificación, buscando puntualizar aquellos que puedan causar daños mayores al sistema.
- ✓ Establecer un posible y prematuro orden de corrección de errores, según la relevancia de estos, para evitar el colapso del sistema.

En el ejemplo mostrado en la tabla anterior, el error de interfaz puede ser el más dañino o nefasto para el sistema, le continúa el error de especificación incompleta, y el que menor impacto tiene es el de violación de uso. Así mismo, el arquitecto podría determinar el orden de estos, si se acordara la necesidad de una corrección inmediata de errores.

La segunda fase tiene como salida la confiabilidad de cada subcaracterística del componente respecto a los errores identificados. El arquitecto a través de esta fase puede medir fácilmente la influencia de los errores en el sistema ya que esta permite:

- ✓ Analizar la confiabilidad desglosada por subcaracterísticas, para identificar aquellas que influyen en la criticidad del error.
- ✓ Detallar los estados por los que atraviesa el componente (camino recorrido), con el fin de particularizar los caminos que pueden conllevar a que el error sea nefasto según la subcaracterística, para el componente.

- ✓ Identificar posibles caminos, para, partiendo del especificado, buscar nuevos recorridos (entre los ya definidos), que afecten en menor grado la confiabilidad del componente, respecto a un error determinado.

La tabla 14 muestra el análisis del modelo de comportamiento dinámico. Este va a tener como salida el valor de la madurez, tolerancia a fallos, y la recuperabilidad del componente ante el error. El ejemplo muestra el camino recorrido por cada error, el peso de este para cada subcaracterística. El valor obtenido va a mostrar cuán maduro, tolerante a fallo, o recuperable puede ser el componente a partir de la ocurrencia de ese error.

Tabla 14. Camino recorrido por cada error del ejemplo. Peso del camino para cada subcaracterística.

Errores	Camino recorrido	Confiabilidad de la Madurez	Confiabilidad de la Tolerancia a fallos	Confiabilidad de la Recuperabilidad
1	E_TC_R	1	1	1
2	E_DA_R	3	2	2
3	E_DNA_FT_R	10	10	8.1

En los resultados anteriores se puede deducir que:

En el primer error, la confiabilidad de la madurez, la tolerancia a fallos, y la recuperabilidad del componente respecto al error es buena, teniendo en cuenta que el peso del camino retornó el mejor valor para la confiabilidad (camino más factible o de menor peso). El componente entra a un estado de tolerancia completa en el que sigue funcionando sin perder funcionalidad y luego se recupera.

Para el segundo error, el componente presenta buena tolerancia ante el fallo debido a que este entra en un estado de degradación en el que mantiene un nivel de ejecución aceptable. La recuperabilidad es buena como en el caso anterior. Sin embargo, la madurez tiene un valor superior, porque a pesar de no llegar al fallo total pierde funcionalidades. Este error presenta una menor confiabilidad (mayor valor), respecto al

primero, esto se debe a que la madurez influyó en el resultado de la misma. Si el componente hubiese tomado a partir del error un estado de tolerancia completa, o de parada de segura, estados en los cuales no pierde funcionalidad, el resultado de la confiabilidad se modificaría positivamente.

En el tercer caso, la confiabilidad de la madurez es la peor, igualmente con la tolerancia ante fallos debido a que según muestra el camino recorrido, el componente llega al fallo total. A pesar de lo expuesto, la recuperabilidad no es la peor, porque logra recuperarse luego del fallo. Para que el error analizado no sea nefasto en el componente, este debe tomar un recorrido que beneficie las subcaracterísticas madurez y tolerancia a fallos. Cualquier camino que no incluya el fallo total, sería suficiente para lograr mejorar la confiabilidad en este caso.

El cálculo de la confiabilidad del componente ante el error, está dado por una fórmula que incluye las confiabilidades de cada subcaracterística, utilizando además el peso del error, que es la salida de la primera fase. Una vez que se obtienen esos valores, se llega al punto donde se requiere el cálculo de la confiabilidad del componente ante ese error (Ver epígrafe 2.2.3). Es importante señalar que el modelo de calidad definido por la investigación no establece prioridades para las subcaracterísticas. En su definición, la confiabilidad de un sistema va a ser confiable según sea maduro, tolerante a fallos y recuperable al mismo nivel. Siguiendo el ejemplo de los puntos anteriores, la tabla 15 ejemplifica los datos del procedimiento antes descrito, el resultado arrojado no es más que la confiabilidad del componente ante el error encontrado. En la medida en que el arquitecto entienda la importancia de cada cálculo, la retroalimentación al proceso de desarrollo de esos datos será mayor. Como este proceso se realizará para cada error será fácil establecer comparaciones entre ellos.

Tabla 15. Cálculo de la confiabilidad del componente producto a los errores

No	C_M	C_{TF}	C_R	P_E	Fórmula	C_{CE}
1.	1	1	1	0.1	$C_{CE} = \frac{1+1+1}{3} \times 0.1$	0.10
2.	3	2	2	0.2		0.47

					$\text{Cce} = \frac{3+2+2}{3} \times 0.2$	
3.	10	10	8	1	$\text{Cce} = \frac{10+10+8.1}{3} \times 1$	9.37

Cuantitativamente queda demostrado que el error más crítico es el tercero, le sigue el segundo y el primer error es el de menor impacto en el componente. A partir de las interpretaciones anteriores se pueden tomar una serie de decisiones que permitan lograr una mejor confiabilidad.

Luego de haber obtenido la confiabilidad del componente para cada error, en la tercera fase se realiza el cálculo de la confiabilidad mediante la utilización del algoritmo K-medias. En la tabla 16 se muestran los resultados de este.

Tabla 16 Resultados de la ejecución del algoritmo K-medias para el ejemplo

Elementos del grupo	Centroide	Cantidad de elementos	Rango para elementos
0.1	0.1	1	Confiable
0.47	0.47	1	Confiable
9.37	9.37	1	No confiable

La confiabilidad del componente va a estar determinada como se describe en el epígrafe 2.2.3, por lo que para este ejemplo quedaría según la representación de la tabla 17.

Tabla 17. Cálculo del valor de la confiabilidad del componente

Centroide1	Centroide1	Centroide 3	Valor de la confiabilidad

0.1	0.47	9.37	9.94

La figura 13 muestra la escala brindada por el modelo matemático para la evaluación del valor de confiabilidad obtenido.

Figura 13. Interpretación del valor obtenido de la confiabilidad del componente.

Analizado el valor de la confiabilidad en la escala propuesta, el arquitecto puede observar que es No confiable.

Lo anteriormente descrito en esta fase posibilita al arquitecto:

- ✓ Establecer comparaciones cuantitativas de las confiabilidades de los errores del componente, con el fin de determinar los errores críticos, y aquellos que afecten en menor grado al componente.
- ✓ Agrupar las confiabilidades según su valor, es decir, los valores pequeños (mejor confiabilidad), los medianos (confiabilidad media), y los mayores (peor confiabilidad).
- ✓ Saber según el centroide que tan bueno, regular, o malo, pueden ser los valores de las confiabilidades obtenidas.
- ✓ Identificar la cantidad de errores que tienen confiabilidades en los 3 diferentes grupos, con el fin de determinar si la mayor cantidad de estos son buenos o viceversa.
- ✓ Establecer un criterio del estado de la confiabilidad del componente a partir de la cantidad de errores y de la criticidad de sus confiabilidades.
- ✓ Poseer el valor cuantitativo de la confiabilidad del componente, valor a través del cual, se puede arribar a una interpretación del estado de confiabilidad del componente.
- ✓ Establecer comparaciones entre confiabilidades de posibles componentes evaluados.

A modo de resumen, en este ejemplo se demuestra que, a pesar de que el componente presenta 2 errores que pueden causar daños menores al sistema, el último error puede hacer que el componente colapse por lo que este eleva el valor de la confiabilidad resultando el componente No confiable.

2.5 Consideraciones generales del modelo definido

El modelo de evaluación definido presenta importantes características que son enumeradas a continuación:

1. Posibilita la evaluación cuantitativa de la confiabilidad de un componente del sistema.
2. Brinda una rápida visión (primera fase), de los defectos más costosos que afectan al componente.
3. Ofrece resultados que detallan la confiabilidad según las subcaracterísticas de la confiabilidad.
4. Determina la criticidad de cada error para el componente.
5. Es completo y robusto, sin la necesidad de datos resultantes de evaluaciones anteriores.

2.6 Conclusiones parciales

- ✓ Para la evaluación de la confiabilidad de Cedrux se requiere adaptar el Modelo de estimación de la confiabilidad de los componentes, manteniendo las buenas prácticas que contiene y modificando aquellas que no se adecuan al sistema.
- ✓ El modelo definido va a poseer una serie de restricciones que son necesarias para el buen funcionamiento del mismo, y que brindarán un mejor entendimiento del modelo.
- ✓ Los resultados provenientes de cada una de las fases del modelo, se encaminan a mostrar las debilidades del sistema. Estas están definidas, de manera que puedan ayudar al equipo de desarrollo a la toma acertada de decisiones arquitectónicas.

Capítulo III: Validación de la solución propuesta

El capítulo tiene como principal objetivo la demostración de la aplicabilidad del modelo matemático para la evaluación de la confiabilidad de un componente de Cedrux. Se mostrarán los resultados de la ejecución del modelo, en función de indicadores demostrativos de la retroalimentación del proceso de desarrollo, y se corroborarán los resultados obtenidos.

3.1 Caracterización del entorno de prueba

La población está constituida por un total de 12 subsistemas de Cedrux. La muestra fue seleccionada teniendo en cuenta los subsistemas que ya están siendo utilizados, de modo que los arquitectos puedan detectar los errores arquitectónicos presentes, y para los cuales la confiabilidad es de gran importancia. Esta queda representada por un total de 9 subsistemas que constituyen un 75 % de la población total. Es importante aclarar, que en la investigación se toma cada subsistema de Cedrux como un componente del sistema. En la tabla 18, se exponen los elementos de la muestra:

Tabla 18 Subsistemas que constituyen la muestra de la investigación

No.	Subsistema
1.	Inventario
2.	Facturación
3.	Activos Fijos
4.	Finanzas
5.	Auditoría
6.	Configuración
7.	Contabilidad
8.	Capital Humano
9.	Marco de trabajo

3.2 Análisis de indicadores

El análisis de los resultados, tras la aplicación del modelo matemático definido para la evaluación de la confiabilidad de Cedrux, se realizó mediante el estudio del comportamiento de una variable y sus indicadores respectivos. El desglose de esta se muestra en la tabla 19.

Tabla 19. Análisis de indicadores

Variable	Dimensión	Indicador	Se propone medir	Medición (Fórmula)	Interpretación del valor
Retroalimentación del proceso de desarrollo.	Información	Exactitud	¿Brinda el modelo información correcta de la confiabilidad al equipo de desarrollo?	No aplica	Sí o No
		Desglose	¿El modelo ofrece información desglosada respecto al estado de las subcaracterísticas de la confiabilidad en el componente?		
		Significación	¿Permite el modelo la obtención de interpretaciones viables que agilicen el proceso de desarrollo?		
Retroalimentación del proceso de desarrollo.	Información	Utilidad	¿Beneficiará el modelo la toma acertada de decisiones arquitectónicas?		
		Comparación	¿Se podrán		

			establecer comparaciones que permitan determinar cuan confiable o no, es un componente respecto a otro?	No aplica	Sí o No

3.3 Datos del entorno de prueba

Los datos para el entorno de prueba fueron tomados a partir de los arquitectos de los subsistemas que se encuentran dentro de la muestra.

Sirviendo como entrada al modelo matemático, se analizaron los subsistemas detectando los errores arquitectónicos en cada uno de ellos. Se estableció la clasificación a la que pertenecen los errores, partiendo de la especificada por el modelo, para arribar a la asignación de pesos a los mismos. De igual manera se identificó en el modelo de comportamiento dinámico, siendo primer paso de la segunda fase, el comportamiento del componente respecto al error (camino recorrido). Es importante señalar que algunos subsistemas presentan errores comunes, por lo que el cálculo de la confiabilidad para estos no se realiza de forma independiente. En las tablas que se muestran a continuación (20- 27) se muestra la ejecución del modelo definido.

Cálculo de la confiabilidad para cada subsistema.

Tabla 20. Cálculo de la confiabilidad para los subsistemas Inventario – Activos Fijos

Subsistema: Inventario - Activos Fijos							
No	**Descripción**	**Peso del Error**	**Camino que recorre**	**Cálculo de las subcaracterísticas**			**Confiabilidad**
				M	**TF**	**R**	
1.	Creación de instancias de objetos globales (integrator, común, etc.) en clases del negocio.	0.1	E-TC-R	1	1	1	0.10
2.	Violación de las capas arquitectónicas. Ejemplo: invocación desde una clase controladora a una clase de acceso a datos; clase controladora ejecutando funciones del	0.1	E-TC-R	1	1	1	0.10

	negocio.						
3.	Incorrecto uso de la Herencia. Existen clases hijas que reproducen funciones de la clase padre, clases padre que no implementan funciones que son utilizadas por sus hijas.	0.2	E-TC-R	1	1	1	0.20
4.	Bajo nivel de reutilización de código para la ejecución de acciones con alto nivel de presencia en el negocio.	0.2	E-PS-R	2.5	3	2.5	0.53
Centroides y confiabilidad del subsistema:				**C1**	**C2**	**C3**	**0.83**
				0.10	0.20	0.53	

Tabla 21. Cálculo de la confiabilidad para el subsistema Auditoría

Subsistema: Auditoría							
No	**Descripción**	**Peso del Error**	**Camino que recorre**	**Cálculo de las subcaracterísticas**			**Confiabilidad**
				M	**TF**	**R**	
1	En el componente analizador existen diferentes clases que contienen los mismos métodos debido al mecanismo de integración.	0.1	E_TC _R	1	1	1	0.10
2.	En la interfaz general existe pérdida de etiquetas, debido a la utilización de alertas, y no el mecanismo que define la arquitectura para ser usado.	0.2	E_DA_R	3	2	2	0.47
3.	En la clase controladora Exportar existe implementada lógica del negocio.	0.1	E_TC_R	1	1	1	0.10
Centroides y confiabilidad del subsistema:				**C1**	**C2**	**C3**	**0.57**
				0.10	0.47	-	

Tabla 22. Cálculo de la confiabilidad para el subsistema Configuración

Subsistema: Configuración							
No	**Descripción**	**Peso del Error**	**Camino que recorre**	**Cálculo de las subcaracterísticas**			**Confiabilidad**
				M	**TF**	**R**	

1.	Cuando una entidad es hija (económicamente) de otra, debe heredar automáticamente de esta, los formatos, actualmente cuando ocurre esta acción el sistema duplica uno de los formatos heredados.	0.8	E_DA_R	3	2	2	1.87
2.	Cuando un documento primario está siendo utilizado por algún subsistema, el componente está permitiendo que el mismo sea modificado, causando inconsistencia en la información.	0.8	E_DNA_R	5	7	5	4.53
3.	Implementaciones de lógica en las clases controladoras.	0.1	E_TC_R	1	1	1	0.10
Centroides y confiabilidad del subsistema:				**C1**	**C2**	**C3**	**6.5**
				0.10	1.87	4.53	

Tabla 23. Cálculo de la confiabilidad para el subsistema Contabilidad

Subsistema: Contabilidad							
No	**Descripción**	**Peso del Error**	**Camino que recorre**	**Cálculo de las subcaracterísticas**			**Confiabilidad**
				M	**TF**	**R**	
1.	Se hace uso del fetchAll en vez del query buscando mejorar el rendimiento de la aplicación por la gran cantidad de datos a cargar de otros componentes del sistema.	0.2	E_TC_R	1	1	1	0.20
Centroides y confiabilidad del subsistema:				**C1**	**C2**	**C3**	**0.20**
				0.20	-	-	

Tabla 24. Cálculo de la confiabilidad para el subsistema Facturación

Subsistema: Facturación							
No	**Descripción**	**Peso del Error**	**Camino que recorre**	**Cálculo de las subcaracterísticas**			**Confiabilidad**
				M	**TF**	**R**	

1.	Violación de las capas arquitectónicas. Ejemplo: invocación desde una clase controladora a una clase de acceso a datos; clase controladora ejecutando funciones del negocio.	0.1	E-TC-R	1	1	1	0.10
2.	Incorrecto uso de la Herencia. Existen clases hijas que reproducen funciones de la clase padre, clases padre que no implementan funciones que son utilizadas por sus hijas.	0.2	E-TC-R	1	1	1	0.20
3.	Bajo nivel de reutilización de código para la ejecución de acciones con alto nivel de presencia en el negocio.	0.2	E-PS-R	2.5	3	2.5	0.53
Centroides y confiabilidad del subsistema:				**C1**	**C2**	**C3**	**0.83**
				0.10	0.20	0.53	

Tabla 25. Cálculo de la confiabilidad para el subsistema Capital Humano

Subsistema: Capital Humano							
No	**Descripción**	**Peso del Error**	**Camino que recorre**	**Cálculo de las subcaracterísticas**			**Confiabilidad**
				M	**TF**	**R**	
1.	Pérdida de etiquetas: al cargar una nueva interfaz y cargar su correspondiente json con sus etiquetas, si se regresa a una interfaz abierta anteriormente no se encuentran las correspondientes a esta.	0.2	E_DA_R	3	2	2	0.47
2.	Clases controladoras conteniendo funcionalidades del negocio.	0.1	E_TC_R	1	1	1	0.10
Centroides y confiabilidad del subsistema:				**C1**	**C2**	**C3**	**0.57**
				0.10	0.47	-	

Tabla 26. Cálculo de la confiabilidad para el subsistema Finanzas

Subsistema: Finanzas					
	Descripción	**Peso**	**Camino**	**Cálculo de las**	**Confiabilidad**

No		del Error	que recorre	subcaracterísticas			
				M	TF	R	
1.	Llamadas directamente a métodos en las clases Domain desde las Controllers o Services sin pasar por las Models.	0.1	E-TC-R	1	1	1	0.10
2.	No se realizan las validaciones mediante el empleo de weaver.xml y validation.xml.	0.1	E-TC-R	1	1	1	0.10
3.	No se tratan los textos de las etiquetas de la interfaz a través del json y en ocasiones cuando se hace no muestra el texto.	0.2	E-DA-R	3	2	2	0.47
4.	Clases Controllers y Services conteniendo funcionalidades de negocio.	0.2	E-DA-R	3	2	2	0.47
Centroides y confiabilidad del subsistema:				**C1**	**C2**	**C3**	**0.57**
				0.10	0.47	-	

Tabla 27. Cálculo de la confiabilidad para el subsistema Marco de trabajo

Subsistema: Marco de Trabajo							
No	**Descripción**	**Peso del Error**	**Camino que recorre**	**Cálculo de las subcaracterísticas**			**Confiabilidad**
				M	**TF**	**R**	
1.	Al existir diferentes conexiones a base de datos para cada subsistema no se puede garantizar la transacionalidad de los procesos que involucran a más de un subsistema.	0.8	E_DNA_PI_R	7	8	6.5	5.73
2.	Las configuraciones de los componentes no están contenidas en los mismos sino en ficheros globales, por lo que un error de configuración de un componente provoca que la aplicación deje de funcionar.	0.2	E_PS_R	2.5	3	2.5	0.53
3.	Existen clases de los componentes que manejan las conexiones a la base de datos, cuando eso es una responsabilidad del marco de trabajo.	0.2	E_TC_R	1	1	1	0.20
4.	Las responsabilidades de las clases						

	están mezcladas existiendo clases de negocio e incluso controladoras implementando consultas o controladoras implementando negocio	0.2	E_TC_R	1	1	1	0.20
Centroides y confiabilidad del subsistema:				**C1**	**C2**	**C3**	**6.46**
				0.20	0.53	5.73	

Resumen del cálculo de las confiabilidades.

La tabla 28 contiene la información desglosada de cada subsistema. Mostrando el centroide de cada grupo en el que se dividen las confiabilidades de los errores presentes, así como, la cantidad de elementos que existen en cada grupo. Por último, muestra el valor de la confiabilidad (suma de los centroides), con su respectiva interpretación para cada subsistema.

Tabla 28. Resumen del cálculo de las confiabilidades

Subsistemas	Grupo(Clúster) 1		Grupo(Clúster) 2		Grupo(Clúster) 3		Confiabilidad	
Centroides No. Elementos	C. Menor	No.Elem	C. Medio	No.Elem	C. Mayor	No.Elem	Valor	Interpretación
Inventario	0.10	2	0.20	1	0.53	1	0.83	Confiable
Facturación	0.10	1	0.20	1	0.53	1	0.83	Confiable
Activos Fijos	0.10	2	0.20	1	0.53	1	0.83	Confiable
Finanzas	0.10	2	0.47	2	-	-	0.57	Confiable
Auditoría	0.10	2	0.47	1	-	-	0.57	Confiable
Configuración	0.10	1	1.87	1	4.53	1	6.5	Poco Confiable
Contabilidad	0.20	1	-	-	-	-	0.20	Confiable
Capital Humano	0.10	1	0.47	1	-	-	0.57	Confiable
Marco de trabajo	0.20	2	0.53	1	5.73	1	6.46	Poco Confiable

Tomando como punto de partida las confiabilidades obtenidas de los componentes evaluados, y mostrados en la tabla anterior, estas fueron organizadas en la escala (Ver figura 14), de modo que quede reflejada la cantidad de subsistemas que pertenecen a cada clasificación (confiable, poco confiable, no confiable). En esta escala se muestra la superioridad en número de los subsistemas confiables (siete), presentando solamente dos poco confiables, y mostrando la inexistencia de subsistemas no confiables.

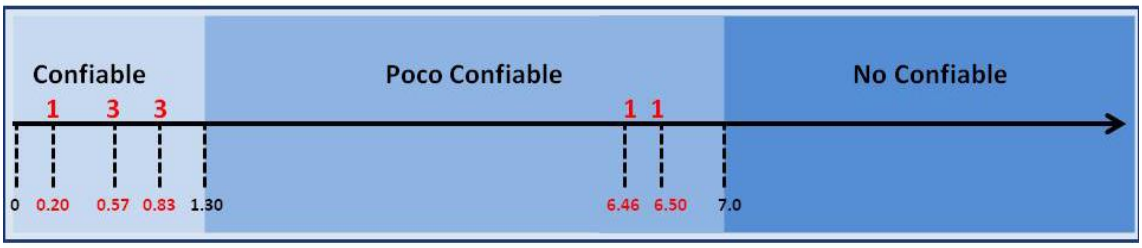

Figura 14. Escala para la evaluación de las confiabilidades obtenidas

Si bien en la escala anterior se mostró una comparación entre la cantidad de subsistemas de cada clasificación, la gráfica mostrada en la figura 15 brinda una comparación de la confiabilidad de cada uno de los subsistemas evaluados. Se puede también observar perfectamente en ella el subsistema de menor valor (confiable), y el de mayor valor, o más crítico.

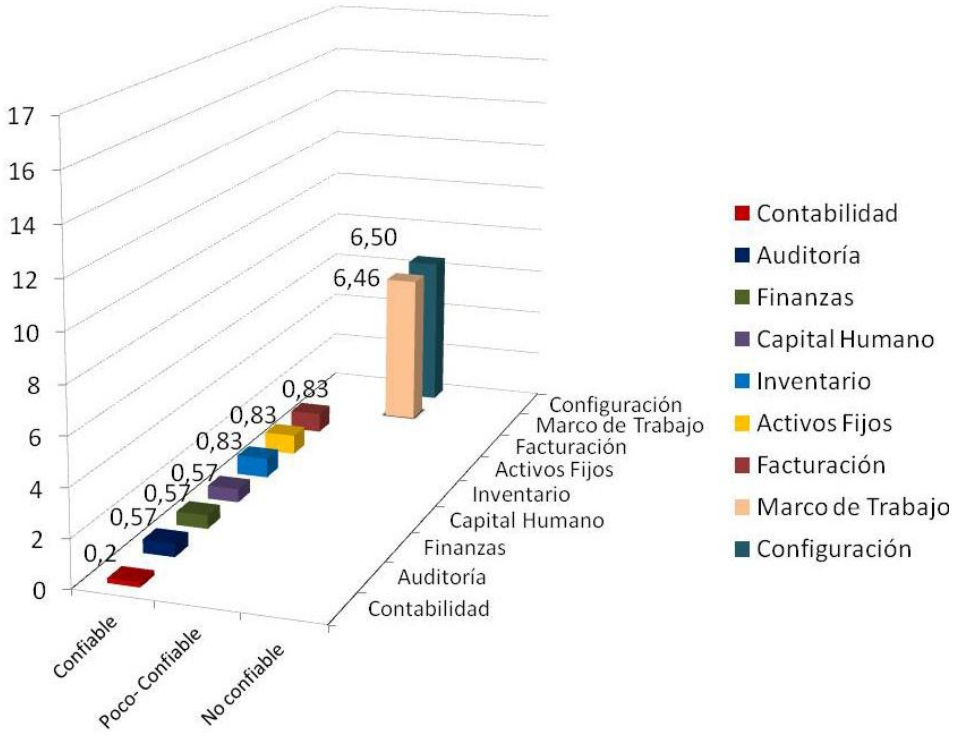

Figura 15. Gráfica representativa de la comparación de confiabilidades de los subsistemas.

3.4 Análisis de resultados

La retroalimentación del proceso de desarrollo, como variable de esta investigación, se verifica mediante una serie de indicadores que toman valores de presencia (Sí), cuando la ejecución del modelo los facilita, y (No) en caso contrario.

Para el análisis de la variable, se va a tomar un subsistema que represente el caso crítico. El subsistema configuración tiene la peor confiabilidad de todos los analizados, este supera en valor a los demás siendo clasificado entre los poco confiables.

Exactitud

Este indicador va a cumplirse en la medida en que el modelo matemático devuelve un valor verídico y exacto de la confiabilidad del subsistema. La confiabilidad para configuración tiene un valor de 6.50, haciéndolo poco confiable. La veracidad de la respuesta del modelo fue corroborada por la opinión del arquitecto del subsistema. Siendo este el que mayor conocimiento tiene respecto al subsistema evaluado (Ver anexo 5).

Desglose

Durante toda la ejecución del modelo, se arrojan resultados que brindan informaciones al arquitecto evaluador. Este indicador va a ser positivo en la medida en que la información pueda ser dividida e interpretada por puntos que sean significativos. El valor de confiabilidad que se obtuvo para el subsistema configuración, previamente se desglosó de la siguiente manera:

- ✓ El peso del error, que determina el impacto de este para el sistema.
- ✓ El valor de cada subcaracterística por error. De modo que el arquitecto puede determinar la subcaracterística que está influenciando la criticidad del error.
- ✓ La confiabilidad de cada error presente en el subsistema, para identificar los errores que están afectando en gran medida la confiabilidad de este, y los que no.

Es decir, que de manera general la evaluación al subsistema configuración cumplió con el indicador, ya que no solo brinda un valor cuantitativo, sino que a lo largo del proceso va desglosando informaciones que contribuyen a la toma de decisiones del equipo de desarrollo.

Significación

Cada fase por la que se transita en el modelo, ofrece resultados que tienen una significación para el arquitecto evaluador. Este indicador se cumple en el subsistema configuración, teniendo en cuenta que cada valor devuelto por el modelo permitió arribar a una interpretación importante. Los valores de las subcaracterísticas tienen su significación en el peso de estas, que está dado según su concepto, y se evalúan en los caminos del modelo de comportamiento dinámico. De igual manera, el modelo ofrece escalas para cada valor cuantitativo al que se arriba.

Utilidad

Del subsistema configuración se determinó la necesidad de corregir los errores de inconsistencia estática del comportamiento. Estos son los que pueden provocar que el subsistema pierda funcionalidades llegando a un estado de ejecución no aceptable. Así mismo, se conoce que la madurez y la tolerancia a fallos son las subcaracterísticas que más afectan la confiabilidad de este. Este indicador se cumple siempre que la información que se obtenga sea útil, interpretando el análisis anterior queda clara la presencia del indicador en el subsistema. La información a la que se va arribando a lo largo de la ejecución del modelo, va a ser relevante y de gran utilidad para la corrección de los errores, y para la futura toma de decisiones arquitectónicas, retribuyendo de esta manera el proceso de desarrollo.

Comparación

La comparación está presente en cada fase del modelo, y a su vez en cada paso dentro de las fases de este. Inicialmente para el subsistema se establecieron comparaciones entre las subcaracterísticas de cada error, entre el peso de los errores presentes en configuración, y luego la confiabilidad de configuración con los restantes subsistemas evaluados. Es importante decir que este indicador es significativo, por las ventajas que puede figurar establecer comparaciones de que puede ser más crítico para el error, para la confiabilidad, o para el sistema en general luego de evaluados los diferentes subsistemas.

Estos indicadores se encuentran explícitos en todos los subsistemas evaluados. De forma general se puede observar su comportamiento en la tabla 29 que resume la presencia de estos en los resultados obtenidos de la aplicación del modelo matemático.

Tabla 29. Verificación del comportamiento de los indicadores

Variable: Retroalimentación del proceso de desarrollo.		
Indicador	**Interpretación del valor**	**Argumentación**
Exactitud	Sí	Los resultados del modelo resultaron valores cuantitativos. Fueron analizados e interpretados por cada arquitecto de los subsistemas evaluados, corroborando los mismos la exactitud de la información obtenida respecto a la confiabilidad.
Desglose	Sí	El modelo visualiza y ofrece al arquitecto, el nivel de confiabilidad de cada subcaracterística en una escala del 0 al 10. Ofrece además, la confiabilidad de cada error antes de llegar a la del sistema en general.
Significación	Sí	Cada valor obtenido durante la ejecución del modelo ofrece una interpretación que da la media al arquitecto de la situación de la confiabilidad. Lo anterior se refleja para las subcaracterísticas, en la confiabilidad de cada error presente en el subsistema, y en la confiabilidad del componente como tal.
Utilidad	Sí	Mediante la escala de interpretación, se define el estado de la confiabilidad de cada uno de los errores que se encuentran en el subsistema, pudiendo identificar los de mayor impacto al sistema en busca de un posible orden de corrección acertado. Ofrece además del valor cuantitativo, la significación de este. De este modo le es posible al equipo de desarrollo arribar al estado real de su confiabilidad, tener la certeza de donde radican los principales errores, para la toma acertada de decisiones arquitectónicas posteriores, y para perfeccionar la confiabilidad actual caso de que esta no sea la mejor, o la deseada.

Comparación	Sí	Durante la ejecución del modelo, se van estableciendo comparaciones beneficiosas para el evaluador, primeramente entre las subcaracterísticas de la confiabilidad, así como entre las confiabilidades de cada uno de los errores. El resultado del modelo, está dado por un valor cuantitativo (además de su interpretación), fácilmente es comparativo con los demás componentes. Teniendo como primicia que a menor valor mayor confiabilidad, este da la medida de la confiabilidad de un componente respecto a otro.

Para confirmar lo expuesto en el análisis de los indicadores se realizaron una serie de entrevistas a los arquitectos de cada uno de los subsistemas evaluados. Cada uno de ellos, analizando previamente el resultado del modelo para su componente, opinó respecto a la veracidad de la respuesta. Esta entrevista permite además, calificar el modelo matemático según su apreciación de los resultados obtenidos (Ver anexos 3-10). En la tabla 30 se resumen por subsistemas los resultados de dicha entrevista.

Tabla 30. Resultado de la entrevista a los arquitectos de los subsistemas evaluados (Anexos 3-10)

Subsistema	Aceptación de la evaluación emitida por el modelo		Categoría asignada al modelo matemático			
	Sí	**No**	**Excelente**	**Bueno**	**Aceptable**	**Malo**
Inventario	X		X			
Facturación	X		X			
Activos Fijos	X		X			
Finanzas	X		X			
Auditoría	X		X			
Configuración	X			X		

Contabilidad	X		X			
Capital Humano	X		X			
Marco de trabajo	X		X			
Total:	100%	0%	88.88%	11.11%		

3.5 Conclusiones Parciales

- ✓ La evaluación de la confiabilidad a los subsistemas de Cedrux mediante el modelo matemático definido, permite la retroalimentación del proceso de desarrollo.
- ✓ La evaluación de la confiabilidad de 9 subsistemas de Cedrux arrojó un total de 7 subsistemas confiables, y 2 poco confiables.
- ✓ La entrevista realizada a los arquitectos de los diferentes subsistemas evaluados de Cedrux, permitió constatar la aprobación del resultado de la evaluación del modelo matemático a los subsistemas en un 100 %, así como la aceptación del modelo matemático con un 88.88 % asignado a la categoría de excelente.

Conclusiones

La investigación realizada permite arribar a las siguientes conclusiones:

- ✓ La evaluación de la confiabilidad en la arquitectura de software es fundamental para la obtención de un Sistema de Planificación de Recursos Empresariales con calidad.
- ✓ La evaluación de la confiabilidad mediante modelos matemáticos, es una técnica cuantitativa, que brinda a los arquitectos un valor exacto del estado de la confiabilidad en la arquitectura del sistema.
- ✓ El Modelo de estimación de la confiabilidad de los componentes de software mediante el aprovechamiento de los modelos arquitectónicos, no se adapta totalmente a las particularidades de Cedrux.
- ✓ El modelo matemático definido para la evaluación de la arquitectura de Cedrux, ofrece resultados que permiten la retribución del proceso de desarrollo de software.
- ✓ La evaluación a nueve subsistemas de Cedrux, y la corroboración de los arquitectos del resultado de los mismos, permiten concluir que el modelo matemático definido brinda información precisa del estado de la confiabilidad del componente evaluado.

Recomendaciones

- ✓ Analizar los resultados (datos) provenientes de diferentes iteraciones del modelo matemático, para, mediante técnicas estadísticas o de inteligencia artificial, se pueda predecir un futuro comportamiento de la confiabilidad de un subsistema de Cedrux.
- ✓ Desarrollar herramientas que sustenten el basamento del modelo matemático definido, con el objetivo de facilitar la utilización de este por parte de los arquitectos del sistema.

Referencias bibliográficas

González,Barbacci, M. 1995. *Quality Attributes.* s.l. : Carnegie Mellon University, 1995.

Barbacci, M., Klein, M., Longstaff, T., & Weinstock, C. 1995. Quality Attributes. [En línea] 1995. http://www.sei.cmu.edu/publications/documents/95.reports/95.tr.021.html.

Barrientos, Justo. 2012. Asegura la eficiencia y la confiabilidad en tu ERP. [En línea] 03 de Enero de 2012. [Citado el: 20 de Abril de 2012.] http://gcg.com.mx/blog/2012/asegura-la-eficiencia-y-la-confiabilidad-en-tu-erp/.

Bass, L., Clements, P., Kazman, R. 1998. *Software Architecture in practice.* s.l. : Addison-Wesley., 1998.

Bass, Len, Clements, Paul and Kazman, Rick. 2003. *Software Architecture in Practice .* s.l. : Addison Wesley, 2003. 0-321-15495-9.

Bosch, J. 2000. *Design & Use of Software Architectures.* 2000.

Brey, Gustavo Andrés., Escobar, Gastón., Passerini, Nicolas., Arias, Juan. 2005. *Arquitectura de Proyectos de IT. Evaluación de Arquitecturas.* Buenos Aires : Universidad Tecnológica Nacional. Facultad Regional de Buenos Aires : Departamento de Sistemas, 2005.

Camacho, Erika. ,Cardeso, Fabio. , Núñes, Gabriel. 2004. *Arquitecturas de Software, Guía de Estudio.* 2004.

Carrascoso, Yoan Arlet., Chaviano,Enrique., Céspedes, Anisleydi. 2009. *Procedimiento para la Evaluación de Arquitecturas de Software basadas en Componentes.* La Habana : s.n., 2009.

Castillo, Jaime F. 2009. *Requerimientos No Funcionales y Arquitectura de Software.* 2009.

Clements, Paul. 1996. *"A Survey of Architecture Description Languages". Proceedings of the International Workshop on Software Specification and Design.* Alemania : s.n., 1996.

Erika Camacho, Fabio Cardeso,Gabriel Núñes. ABRIL – 2004. *Arquitecturas de Software, Guía de Estudio.* ABRIL – 2004.

Extremadura, Universidad de. 2008. *Tolerancia a fallos.* 2008.

Fernández, Elmer. A. 2010. *Clustering (Agrupamiento).* 2010.

Fernández, Oscar M. ,García León, Delba. ,Beltrán Benavides, Alfa. 1995. *Un enfoque actual sobre la calidad del software.* 1995.

Gómez, Omar S. 1997. *Evaluando la arquitectura de software.* 1997.

Gómez, Omar Salvador Gómez. 2005. *Proceso de evaluación para arquitecturas de software usadas en el sector empresarial (PEASSE).* 2005.

Góngora, Asnier Enrique. 2011. *Catálogo automatizado de métricas de calidad para evaluar los productos en las pruebas.* 2011.

González, Mario. 2008. *Arquitectura de software.* 2008.

Ibañes, Juaquim. 2010. Gestión de requerimientos I: Definición. Los stakeholders. [En línea] Octubre de 2010.

IEEE. 1990. *IEEE Standard Glossary of Software Engineering Terminology.* 1990.

ISO/IEC. 1998. Information Technology – Software Product Quality – Part 1:. [En línea] 1998. http://www.usability.serco.com/trump/resources/standards.htm#9126-1.

Kazman, R, Clements, P y Klein, M. 2001. *Evaluating Software Architectures.* s.l. : Addison Wesley, 2001.

Kazman, R., Clements, P., Klein, M. 2001. *Evaluating Software Architectures.* s.l. : Addison Wesley, 2001.

Martínez, Octavio R. Lara. 2006. *Sistemas de Gestión: " Cómo Integrar Procesos y Automatizar Actividades de su Negocio ".* 2006.

Meier, J D. 2008. *Application Architecture Guide 2.0. : Microsoft Press.* 2008.

Milán, Yisét. 2011. *Propuesta de escenarios arquitectónicos y atributos de calidad para la evaluación arquitectónica del sistema Cedrux.* Ciudad de la Habana : Universidad de las Ciencias Informaticas, 2011.

Oficina Nacional de Normalización. 2005. *Ingeniería de software.Calidad del producto. Parte I: Modelo de la calidad ISO/IEC 9126-1:2001, IDT).* Ciudad de La Habana. : s.n., 2005. 35.080.

Panovski, Gregor. 2008. *Product Software Quality.* 2008.

Pressman, R. 2002. *Ingeniería de Software. Un Enfoque Práctico.* 2002.

Pressman, R. 2002. *Ingeniería de Software. Un Enfoque Práctico.* 2002.

Reynoso, Carlos Billy. 2004. *Introducción a la Arquitectura de Software Versión 1.0.* Universidad de Buenos Aires : s.n., 2004.

Roshandel, Roshanak., Banerjee, Somo. ,Cheung, Leslie., Medvidovic, Nenad., Golubchik, Leana. 2006. *Estimating Software Component Reliability by Leveraging Architectural Models.* 2006.

Scalone, Lic. Fernanda. 2006. *Estudio comparativo de los modelos y éstandares de calidad del software.* Buenos Aires : Universidad Tecnológica Nacional de Buenos Aires, 2006.

Stivens, Yuliet., Bencomo, Marcos. 2009. *Diseño e Implementación de la Arquitectura.* 2009.

Tekinerdogan, Bedir., Sözer, Hasan. , Aksit, Mehmet. 2005. *Software Architecture Reliability Analysis using Failure Scenarios.* 2005.

Vaziri, Reza., Javidan, Reza. 2011. *A Bayesian Based Model for Evaluation of Software Architecture Reliability.* 2011.

Vera, Angelo Benvenuto. 2006. Implementación de Sistemas ERP, su impacto en la gestión de la empresa e integración con otras TIC. [En línea] 2006.

Anexos

Anexo 1. Modelo de la encuesta I realizada a arquitectos de Cedrux.

Responda a cada una de las siguientes preguntas:

1. ¿Se ha realizado alguna evaluación a la Arquitectura en el proyecto al que pertenece?

   Si ___ No ___

2. ¿Conoce y utiliza los métodos de evaluación de arquitectura? Menciónelos.

   _____ _____ _____

   _____ _____ _____

3. Diga qué técnicas de las que se muestran a continuación, utiliza para realizar la evaluación arquitectónica de su proyecto.

   Escenarios ___ Simulación ___ Experiencia ___

   Modelos Matemáticos ___ **Otra** ___

4. ¿Sabe usted el momento factible para la realización de la evaluación de la arquitectura?

   Si ___ No ___

5. ¿Opina usted que la confiabilidad es un atributo de calidad relevante durante la evaluación de la arquitectura?

   Si ___ No ___

6. ¿Tiene conocimientos de los métodos y técnicas a aplicar para evaluar la confiabilidad del software?

   Si ___ No ___

7. Teniendo en cuenta que, mediante de las técnicas cuantitativas se obtienen resultados numéricos para arribar a interpretaciones del estado del sistema, mientras que las técnicas cualitativas retornan generalmente SI o No, sin

mayor nivel de detalle. Cree usted, para agilizar y retribuir el proceso de desarrollo del software, es factible realizar un proceso de evaluación de la confiabilidad del sistema mediante técnicas:

_____ Cuantitativas O _____Cualitativas.

8. Considera usted que mediante la evaluación cuantitativa del software puede obtener resultados que reflejen cuán satisfactorio es el diseño arquitectónico.

   Si ___ No ___

Anexo 2. Modelo de la encuesta II realizada a arquitectos de Cedrux.

En el centro CEIGE se encuentra en análisis una propuesta de evaluación de la **Confiabilidad** de los componentes del sistema Cedrux. Tome en cuenta que la Confiabilidad para la presente propuesta se asume tal y como la define la norma de calidad ISO/IEC 9126-2005: La capacidad del producto de software para mantener un nivel de ejecución especificado cuando se usa bajo las condiciones especificadas. Esta norma define el análisis de Confiabilidad a partir de:

Madurez: Capacidad del producto de software de evitar un fallo total como resultado de haberse producido un fallo del software.

Tolerancia ante fallos: Capacidad del producto de software de mantener un nivel de ejecución o desempeño especificado en caso de fallos del software o de infracción de su interfaz especificada.

Recuperabilidad: Capacidad del producto de software de restablecer un nivel de ejecución especificado y recuperar los datos directamente afectados en caso de fallo total.

Para dicha evaluación se está proponiendo la utilización de una técnica de evaluación cuantitativa. El modelo matemático definido a tales efectos, en su basamento, hace una clasificación de posibles errores arquitectónicos. Estos errores van a tener un peso o costo para el componente de manera que a mayor peso, mayor será el impacto negativo sobre la **Confiabilidad** del sistema.

Resulta necesario entonces adecuar el modelo matemático a las particularidades de los componentes y su desarrollo bajo el modelo de construcción de Cedrux, por lo que se requiere que usted responda las siguientes preguntas.

Pregunta 1

Analice la clasificación de errores arquitectónicos para el cálculo de la confiabilidad que se muestra en la figura 1 y responda de esta clasificación las siguientes preguntas:

a) ¿Cree usted que la clasificación se adapta a los compontes de Cedrux? Marque *Si*, *No, P (parcialmente)* al lado del error.
b) Incluya los errores faltantes que sean necesarios en su opinión, y proponga las modificaciones que considere a los errores marcados con *P*.

Errores topológicos: Tienden a ser globales en la arquitectura y se refieren a aspectos relacionados con la configuración de componentes y conectores en el sistema. A menudo son resultado de la violación de las restricciones impuestas por los estilos arquitectónicos. Los errores topológicos son de naturaleza direccional o estructural:

_____ La dirección: Específica si la comunicación necesaria por el estilo arquitectónico es violada. Un ejemplo de este error es cuando en una arquitectura cliente-servidor el servidor solicita un servicio a partir de un cliente.

Estructura: En su naturaleza se dividen en Violaciones de uso y Especificación incompleta.

_____ Violación de uso: Un ejemplo de una violación de uso es cuando un enlace de comunicación entre dos componentes está ausente, o, alternativamente, cuando una comunicación relación entre los componentes existe donde no debe estar presente.

_____ Especificación incompleta: Se manifiesta cuando hay principalmente información insuficiente para especificar las propiedades de los componentes del modelo arquitectónico y conectores. (Errores internos en los componentes por pobre delimitación de sus responsabilidades)

Inconsistencias de comportamiento: Se relacionan con la información arquitectónica local a un componente, o un servicio de un conector un conjunto de componentes que interactúan. Se refieren a los desajustes entre las interfaces, los comportamientos estáticos, o protocolo de interacción componentes. Por su naturaleza se dividen en:

_____ Interfaz: Un defecto de interfaz ocurre cuando las etiquetas de servicios proporcionados y/o requeridos entre dos componentes no coinciden.

_____Inconsistencia estática del comportamiento: Revela desajustes entre las pre y post-condiciones proporcionadas correspondientemente y servicios que se requieren en dos componentes.

_____Inconsistencia en el protocolo: Pone de manifiesto los errores de interacción de protocolos entre los componentes.

Pregunta 2

Asigne según su opinión un costo a cada error en un rango del 1 al 10, de forma tal que el que obtenga el costo superior sea el de mayor impacto negativo sobre el sistema, y viceversa. Trate en la medida de lo posible de no asignar el mismo valor a 2 errores con el fin de hacer notar la diferencia en sus respectivas importancias. Por favor asigne costo a los errores incluidos por usted en la pregunta anterior.

_____La dirección

_____Violación de uso

_____Especificación incompleta

_____Interfaz

_____Inconsistencia estática del comportamiento

_____Inconsistencia en el protocolo

Anexo 3. Modelo de entrevista al arquitecto de los subsistemas Inventario y Activos Fijos de Cedrux, para validar el resultado del modelo matemático.

Entrevista al arquitecto de los subsistemas Inventario y Activos Fijos

Resultado del Modelo matemático para la evaluación de la confiabilidad de Cedrux.

Subsistema: Inventario - Activos Fijos									
No	Descripción de los errores detectados	Clasificación	Peso del Error	Camino que recorre	Cálculo de las subcaracterísticas			Confiabilidad	
					M	TC	R	Valor	Interpretación
1.	Creación de instancias de objetos globales (integrator, común, etc.) en clases del negocio.	Violación de uso	0.1	E-TC-R	1	1	1	0.10	Confiable
2.	Violación de las capas arquitectónicas. Ejemplo: invocación desde una clase controladora a una clase de acceso a datos; clase controladora ejecutando funciones del negocio.	Violación de uso	0.1	E-TC-R	1	1	1	0.10	Confiable
3.	Incorrecto uso de la Herencia. Existen clases hijas que reproducen funciones de la clase padre, clases padre que no implementan funciones que son utilizadas por sus hijas.	Especificación incompleta	0.2	E-TC-R	1	1	1	0.20	Confiable
4.	Bajo nivel de reutilización de código para la ejecución de acciones con alto nivel de presencia en el negocio.	Especificación incompleta	0.2	E-PS-R	2.5	3	2.5	0.53	Confiable
				Confiabilidad del subsistema:				0.83	Confiable

Resumen e interpretación del resultado

Los errores que están influyendo en la confiabilidad de los subsistemas evaluados se ajustan a las siguientes clasificaciones:

1. Violación de uso.
2. Especificación incompleta.

Los pesos asignados a estas clasificaciones según su impacto en el sistema son 0.1 y 0.2 respectivamente, lo que determina la superioridad en impacto de los de especificación incompleta, respecto a los de violación de uso. Lo anterior, y el resultado de las confiabilidades iguales de ambos, e inferiores además a las de los errores de especificación incompleta, ofrecen el orden de importancia de corrección. Sin embargo, al tener estos errores los pesos mínimos de la clasificación, no son capaces de provocar un colapso del sistema.

Para los tres primeros errores presentes en los subsistemas, estos toman un estado de tolerancia completa ante el error, es decir no pierden información, ni funcionalidades. Para cada uno de ellos las subcaracterísticas tienen el mejor valor posible (menor), y este puede ser el mejor comportamiento de los subsistemas respecto a un error por lo que son maduros, tolerantes a fallos, y recuperables. No existe la posibilidad de un estado de paro o de fallo total.

El último error pierde confiabilidad debido a que, a pesar de no perder funcionalidades, ni información, los subsistemas se detienen de forma segura. La madurez y la recuperabilidad son menores que en el caso anterior, pero la subcaracterística que más afecta la confiabilidad es la tolerancia a fallos. Si los subsistemas a partir del error, tomaran un estado de tolerancia completa, la confiabilidad aumentaría significativamente.

Según lo ya analizado, los subsistemas no contienen errores arquitectónicos que puedan hacer que pierdan funcionalidades o lleguen a un estado de fallo total. El resultado final de la confiabilidad de los subsistemas es de 0.83, lo que los ubica según la escala planteada por el modelo en un subsistema **Confiable**.

Pregunta al arquitecto del subsistema Inventario y Activos Fijos:

Cuántos años de experiencia tiene en el rol de arquitecto: 3.
En cuántos subsistemas ha desempeñado su rol: 3.
Cuáles: Inventario, Facturación, Activos Fijos.

1. De acuerdo a su juicio, y analizando las deducciones obtenidas mediante la ejecución del modelo matemático, coincide usted con la respuesta de este modelo para el subsistema Inventario y Activos Fijos:

   X Si ___No ___Parcialmente.

2. Según su opinión, qué categoría le otorga de manera general a la propuesta:

   X Excelente ___ Bueno ___ Aceptable ___ Malo

Ing. Javier Ramirez Hernández

Anexo 4. Modelo de entrevista al arquitecto del subsistema Capital Humano de Cedrux, para validar el resultado del modelo matemático.

Entrevista al arquitecto del subsistema Capital Humano

Resultado del Modelo matemático para la evaluación de la confiabilidad de Cedrux.

Subsistema: Capital Humano									
No	Descripción de los errores detectados	Clasificación	Peso del Error	Camino que recorre	Cálculo de las subcaracteristicas			Confiabilidad	
					M	TC	R	Valor	Interpretación
1.	Pérdida de etiquetas: al cargar una nueva interfaz y cargar su correspondiente json con sus etiquetas, si se regresa a una interfaz abierta anteriormente no se encuentran las correspondientes a esta.	Especificación incompleta	0.2	E_DA_R	3	2	2	0.47	Confiable
2.	Clases controladoras conteniendo funcionalidades del negocio.	Violación de uso	0.1	E_TC_R	1	1	1	0.10	Confiable
							Confiabilidad del subsistema:	0.57	Confiable

Resumen e interpretación del resultado

Los errores que están influyendo en la confiabilidad del subsistema evaluado se ajustan a las siguientes clasificaciones:

1. Violación de uso.
2. Especificación incompleta.

Los pesos asignados a estas clasificaciones según su impacto en el sistema son 0.1 y 0.2 respectivamente, lo que determina la superioridad en impacto de el de especificación incompleta, respecto al de de violación de uso. Lo anterior, y el resultado de las confiabilidades ambos errores, ofrecen el orden de importancia de corrección. Sin embargo, al tener estos errores los pesos mínimos de la clasificación, no son capaces de provocar un colapso del sistema.

Para el primer error presente (especificación incompleta), aunque este se degrada el subsistema sigue funcionando bajo un nivel de ejecución aceptable. Este tiene buena tolerancia a fallos y recuperabilidad, pero la subcaracterística influyente es la madurez. Si a partir de ese error el subsistema hubiese entrado en un estado de tolerancia completa, la confiabilidad mejoraría considerablemente.

Frente al segundo error (violación de uso) el subsistema toma un estado de tolerancia completa ante el error, es decir no pierde información, ni funcionalidad. Las subcaracterísticas tienen el mejor valor posible (menor), y este puede ser el mejor comportamiento del subsistema respecto a un error por lo que es maduro, tolerante a fallos, y recuperable. No existe la posibilidad de un estado de paro o de fallo total.

Según lo ya analizado, el subsistema no contiene errores arquitectónicos que puedan provocar que este llegue a un nivel de ejecución no aceptable, o a un estado de fallo total. El resultado final de la confiabilidad del subsistema es de 0.57, lo que lo ubica según la escala planteada por el modelo en un subsistema **Confiable**.

Pregunta al arquitecto del subsistema Capital Humano:

Cuántos años de experiencia tiene en el rol de arquitecto: 1.

En cuántos subsistemas ha desempeñado su rol: 1.

Cuáles: Capital Humano.

1. De acuerdo a su juicio, y analizando las deducciones obtenidas mediante la ejecución del modelo matemático, coincide usted con la respuesta de este modelo para el subsistema Capital Humano:

   X Si ___No ___Parcialmente.

2. Según su opinión, qué categoría le otorga de manera general a la propuesta:

   X Excelente ___ Bueno ___ Aceptable ___ Malo.

William González Obregón.

Anexo 5. Modelo de entrevista al arquitecto del subsistema Configuración de Cedrux, para validar el resultado del modelo matemático.

Entrevista al arquitecto del subsistema Configuración

Resultado del Modelo matemático para la evaluación de la confiabilidad de Cedrux.

Subsistema: Configuración									
No	Descripción de los errores detectados	Clasificación	Peso del Error	Camino que recorre	Cálculo de las subcaracterísticas			Confiabilidad	
					M	TC	R	Valor	Interpretación
1	Cuando una entidad es hija (económicamente) de otra, debe heredar automáticamente de esta, los formatos, actualmente cuando ocurre esta acción el sistema duplica uno de los formatos heredados.	Inconsistencia estática del comportamiento	0.8	E_DA_R	3	2	2	1.87	Poco Confiable
2.	Cuando un documento primario está siendo utilizado por algún subsistema, el componente está permitiendo que el mismo sea modificado, causando inconsistencia en la información.	Inconsistencia estática del comportamiento	0.8	E_DNA_R	5	7	5	4.53	Poco Confiable
3.	Implementaciones de lógica en las clases controladoras.	Violación de uso	0.1	E_TC_R	1	1	1	0.10	Confiable
							Confiabilidad del subsistema:	6.5	Poco Confiable

Resumen e interpretación del resultado

Los errores que están influyendo en la confiabilidad del subsistema evaluado se ajustan a las siguientes clasificaciones:

1. Violación de uso.
2. Inconsistencia estática del comportamiento

Los pesos asignados a estas clasificaciones según su impacto en el sistema son 0.1 y 0.8 respectivamente, lo que determina la superioridad en impacto del error de los errores de Inconsistencia estática del comportamiento, respecto al de violación de uso. Lo anterior, y el resultado de las confiabilidades iguales de los errores de mayor peso, y superiores además, a la del error de violación de uso, ofrecen el orden de importancia de corrección.

Sin embargo, a pesar de que el error de violación de uso es el último en el orden de corrección, y no es capaz de provocar un colapso del sistema, los de Inconsistencia estática del comportamiento pueden hacer que el sistema pierda funcionalidades llegando hasta un nivel de ejecución no aceptable.

El error de violación de uso presente, hace que el subsistema tome un estado de tolerancia completa ante el error, es decir no pierde información, ni funcionalidades. Para este error las subcaracterísticas tienen el mejor valor posible (menor), y este puede ser el mejor comportamiento del subsistema respecto a un error por lo que es maduro, tolerante a fallos, y recuperable. No existe la posibilidad de un estado de paro o de fallo total.

Los errores de inconsistencia estática del comportamiento, por el contrario hacen que el subsistema pierda funcionalidades. A pesar que el primer error permite una ejecución aceptable, no pasa lo mismo en el segundo caso, en el que el componente se degrada al punto de mantenerse a un nivel de ejecución no aceptable. Para el primer error, la recuperabilidad y la tolerancia a fallos tienen peor confiabilidad que en el caso anterior, sin embargo, la madurez en este caso es la subcaracterística que más afecta al subsistema. Si a partir del error, el subsistema tomara un estado de tolerancia completa, la madurez aumentaría significativamente. Para el segundo caso, la confiabilidad del componente respecto al error es aún más crítica, y l tolerancia a fallos es la subcaracterística que más afecta al subsistema. Si este a partir del error hubiese entrado en un estado de degradación aceptable o en el mejor de los casos de tolerancia completa, la confiabilidad se modificaría para mostrar mejores resultados.

Según lo ya analizado, el subsistema contiene errores arquitectónicos que puedan hacer que este no siga funcionando bajo un nivel de ejecución aceptable, siendo el peor de los errores evaluados en el. Eso puede provocar daños importantes al sistema, aunque no llegue a un estado de fallo total. El resultado final de la confiabilidad del subsistema es de 6.5, lo que lo ubica según la escala planteada por el modelo en un subsistema **Poco confiable**.

Pregunta al arquitecto del subsistema Configuración:

Cuántos años de experiencia tiene en el rol de arquitecto: 1.
En cuántos subsistemas ha desempeñado su rol: 2.
Cuáles: Configuración, Estructura y Composición

1. De acuerdo a su juicio, y analizando las deducciones obtenidas mediante la ejecución del modelo matemático, coincide usted con la respuesta de este modelo para el subsistema Configuración:

   X Si ___No ___Parcialmente.

2. Según su opinión, qué categoría le otorga de manera general a la propuesta:
   ___ Excelente X Bueno ___ Aceptable ___ Malo.

Eliecer Cabrera Casas.

Anexo 6. Modelo de entrevista al arquitecto del subsistema Facturación de Cedrux, para validar el resultado del modelo matemático.

Entrevista al arquitecto del subsistema Facturación

Resultado del Modelo matemático para la evaluación de la confiabilidad de Cedrux.

Subsistema: Facturación									
No	Descripción de los errores detectados	Clasificación	Peso del Error	Camino que recorre	Cálculo de las subcaracteristicas			Confiabilidad	
					M	TC	R	Valor	Interpretación
1.	Violación de las capas arquitectónicas. Ejemplo: invocación desde una clase controladora a una clase de acceso a datos; clase controladora ejecutando funciones del negocio.	Violación de uso	0.1	E-TC-R	1	1	1	0.10	Confiable
2.	Incorrecto uso de la Herencia. Existen clases hijas que reproducen funciones de la clase padre, clases padre que no implementan funciones que son utilizadas por sus hijas.	Especificación incompleta	0.2	E-TC-R	1	1	1	0.20	Confiable
3.	Bajo nivel de reutilización de código para la ejecución de acciones con alto nivel de presencia en el negocio.	Especificación incompleta	0.2	E-PS-R	2.5	3	2.5	0.53	Confiable
							Confiabilidad del subsistema:	0.83	Confiable

Resumen e interpretación del resultado

Los errores que están influyendo en la confiabilidad del subsistema evaluado se ajustan a las siguientes clasificaciones:

1. Violación de uso.
2. Especificación incompleta.

Los pesos asignados a estas clasificaciones según su impacto en el sistema son 0.1 y 0.2 respectivamente, lo que determina la superioridad en impacto de los de especificación incompleta, respecto a los de violación de uso. Lo anterior, y el resultado de las confiabilidades iguales de los errores de especificación incompleta, y superiores además a la del error de violación de uso, ofrecen el orden de importancia de corrección. Sin embargo, al tener estos errores los pesos mínimos de la clasificación, no son capaces de provocar un colapso del sistema.

Para los dos primeros errores presentes en el subsistema, este toma un estado de tolerancia completa ante el error, es decir no pierde información, ni funcionalidad. Para cada error, las subcaracterísticas tienen el mejor valor posible (menor), y este puede ser el mejor comportamiento del subsistema respecto a un error por lo que es maduro, tolerante a fallos, y recuperable. No existe la posibilidad de un estado de paro o de fallo total.

El último error pierde confiabilidad debido a que, a pesar de no perder funcionalidades, ni información, el subsistema se detiene de forma segura. La madurez y la recuperabilidad son peores que en el caso anterior, pero la subcaracterística que más afecta la confiabilidad es la tolerancia a fallos. Si el subsistema a partir del error, tomara un estado de tolerancia completa, la confiabilidad aumentaría significativamente.

Según lo ya analizado, el subsistema no contiene errores arquitectónicos que puedan hacer que pierda funcionalidades o llegue a un estado de fallo total. El resultado final de la confiabilidad del subsistema es de 0.83, lo que lo ubica según la escala planteada por el modelo en un subsistema **Confiable**.

Pregunta al arquitecto del subsistema Facturación:

Cuántos años de experiencia tiene en el rol de arquitecto: 3.
En cuántos subsistemas ha desempeñado su rol: 3.
Cuáles: Inventario, Facturación, Activos Fijos.

1. De acuerdo a su juicio, y analizando las deducciones obtenidas mediante la ejecución del modelo matemático, coincide usted con la respuesta de este modelo para el subsistema Facturación:

   X Si ___No ___Parcialmente.

2. Según su opinión, qué categoría le otorga de manera general a la propuesta:

   X Excelente ___ Bueno ___ Aceptable ___ Malo.

Ing. Javier Ramirez Hernández

Anexo 7. Modelo de entrevista al arquitecto del subsistema Finanzas de Cedrux, para validar el resultado del modelo matemático.

Entrevista al arquitecto del subsistema Finanzas

Resultado del Modelo matemático para la evaluación de la confiabilidad de Cedrux.

Subsistema: Finanzas									
No	Descripción de los errores detectados	Clasificación	Peso del Error	Camino que recorre	Cálculo de las subcaracterísticas			Confiabilidad	
					M	TC	R	Valor	Interpretación
1.	Llamadas directamente a métodos en las clases Domain desde las Controllers o Services sin pasar por las Models.	Violación de uso	0.1	E-TC-R	1	1	1	0.10	Confiable
2.	No se realizan las validaciones mediante el empleo de weaver.xml y validation.xml.	Violación de uso	0.1	E-TC-R	1	1	1	0.10	Confiable
3.	No se tratan los textos de las etiquetas de la interfaz a través del json y en ocasiones cuando se hace no muestra el texto.	Especificación incompleta	0.2	E-DA-R	3	2	2	0.47	Confiable
4.	Clases Controllers y Services conteniendo funcionalidades de negocio.	Especificación incompleta	0.2	E-DA-R	3	2	2	0.47	Confiable
				Confiabilidad del subsistema:				0.57	Confiable

Resumen e interpretación del resultado

Los errores que están influyendo en la confiabilidad del subsistema evaluado se ajustan a las siguientes clasificaciones:

1. Violación de uso.
2. Especificación incompleta.

Los pesos asignados a estas clasificaciones según su impacto en el sistema son 0.1 y 0.2 respectivamente, lo que determina la superioridad en impacto de los de especificación incompleta, respecto a los de violación de uso. Lo anterior, y el resultado de las confiabilidades iguales de los errores de especificación incompleta, y superiores además a las de los errores de violación de uso, ofrecen el orden de importancia de corrección. Sin embargo, al tener estos errores los pesos mínimos de la clasificación, no son capaces de provocar un colapso del sistema.

Para los dos primeros errores presentes en el subsistema, este toma un estado de tolerancia completa ante el error, es decir no pierde información, ni funcionalidad. Para cada error, las subcaracterísticas tienen el mejor valor posible (menor), y este puede ser el mejor comportamiento del subsistema respecto a un error por lo que es maduro, tolerante a fallos, y recuperable. No existe la posibilidad de un estado de paro o de fallo total.

Los últimos errores pierden confiabilidad debido a que, a pesar de que sigue funcionando bajo un nivel de ejecución aceptable, el subsistema se degrada perdiendo funcionalidades. La recuperabilidad y la tolerancia a fallos tienen peor confiabilidad que en el caso anterior, sin embargo, la madurez en este caso es la subcaracterística que más afecta al subsistema. Si a partir de estos errores, el subsistema tomara un estado de tolerancia completa, la madurez aumentaría significativamente.

Según lo ya analizado, el subsistema no contiene errores arquitectónicos que puedan hacer que este siga funcionando bajo un nivel de ejecución no aceptable o llegue a un estado de fallo total. El resultado final de la confiabilidad del subsistema es de 0.57, lo que lo ubica según la escala planteada por el modelo en un subsistema **Confiable**.

Pregunta al arquitecto del subsistema Finanzas:

Cuántos años de experiencia tiene en el rol de arquitecto: 1 año.

En cuántos subsistemas ha desempeñado su rol: 1 finanzas.

Cuáles: finanzas.

De acuerdo a su juicio, y analizando las deducciones obtenidas mediante la ejecución del modelo matemático, coincide usted con la respuesta de este modelo para el subsistema Finanzas:

√ Si ___No ___Parcialmente.

Según su opinión, qué categoría le otorga de manera general a la propuesta:

1 Excelente ___ Bueno ___ Aceptable ___ Malo

Ing. Pedro Antonio Torres Salas

Anexo 8. Modelo de entrevista a un arquitecto del Marco de trabajo de Cedrux, para validar el resultado del modelo matemático.

Entrevista a un arquitecto del Marco de trabajo

Resultado del Modelo matemático para la evaluación de la confiabilidad de Cedrux.

Subsistema: Marco de trabajo									
No	Descripción de los errores detectados	Clasificación	Peso del Error	Camino que recorre	Cálculo de las subcaracterísticas			Confiabilidad	
					M	TC	R	Valor	Interpretación
1.	Al existir diferentes conexiones a base de datos para cada subsistema no se puede garantizar la transacionalidad de los procesos que involucran a más de un subsistema.	Inconsistencia estática del comportamiento	0.8	E_DNA_PI_R	7	8	6.5	5.73	Poco Confiable
2.	Las configuraciones de los componentes no están contenidas en los mismos sino en ficheros globales, por lo que un error de configuración de un componente provoca que la aplicación deje de funcionar.	Especificación incompleta	0.2	E_PS_R	2.5	3	2.5	0.53	Confiable
3.	Existen clases de los componentes que manejan las conexiones a la base de datos, cuando eso es una responsabilidad del marco de trabajo.	Especificación incompleta	0.2	E_TC_R	1	1	1	0.20	Confiable
4.	Las responsabilidades de las clases están mezcladas existiendo clases de negocio e incluso controladoras implementando consultas o controladoras implementando negocio	Especificación incompleta	0.2	E_TC_R	1	1	1	0.20	Confiable
				Confiabilidad del subsistema:				**6.46**	**Poco Confiable**

Resumen e interpretación del resultado

Los errores que están influyendo en la confiabilidad del subsistema evaluado se ajustan a las siguientes clasificaciones:

1. Especificación incompleta.
2. Inconsistencia estática del comportamiento.

Los pesos asignados a estas clasificaciones según su impacto en el sistema son 0.2 y 0.8 respectivamente, lo que determina la superioridad en impacto del error de Inconsistencia estática del comportamiento, respecto a los de especificación incompleta. Lo anterior, y el resultado de las confiabilidades ofrecen el orden de importancia de corrección.

El primer error (inconsistencia estática del comportamiento) puede hacer que el componente pierda funcionalidades llegando a un nivel de ejecución no aceptable, pasa a una parada insegura donde pierde además información y luego se recupera. Para este error en específico el componente es poco confiable. Si por el contrario el componente degradara sin paro inseguro y se mantuviera aceptablemente la confiabilidad sería muy superior. El segundo error pierde confiabilidad debido a que, a pesar de no perder funcionalidades, ni información, el subsistema se detiene de forma segura. La madurez y la recuperabilidad no son las mejores, pero la subcaracterística que más afecta la confiabilidad es la tolerancia a fallos. Si el subsistema a partir del error, tomara un estado de tolerancia completa, la confiabilidad aumentaría significativamente. Los dos últimos errores presentes, el subsistema toma un estado de tolerancia completa ante el error, es decir no pierde información, ni funcionalidad. Para cada error, las subcaracterísticas tienen el mejor valor posible (menor), y este puede ser el mejor comportamiento del subsistema respecto a un error por lo que es maduro, tolerante a fallos, y recuperable. No existe la posibilidad de un estado de paro o de fallo total.

Según lo ya analizado, el subsistema contiene errores arquitectónicos que pueden provocar que pierda funcionalidades, parando inseguramente, aunque no llegue a un estado de fallo total. El resultado final de la confiabilidad del subsistema es de 6.46, lo que lo ubica según la escala planteada por el modelo en un subsistema **Poco confiable**.

Pregunta a un arquitecto del Marco de trabajo: Javier Ruiz Durán

Cuántos años de experiencia tiene en el rol de arquitecto: 2.

En cuántos subsistemas ha desempeñado su rol: 2.

Cuáles: Seguridad Trazas.

De acuerdo a su juicio, y analizando las deducciones obtenidas mediante la ejecución del modelo matemático, coincide usted con la respuesta de este modelo para el subsistema Marco de trabajo:

X Si ___No ___Parcialmente.

Según su opinión, qué categoría le otorga de manera general a la propuesta:

X Excelente ___ Bueno ___ Aceptable ___ Malo

Anexo 9. Modelo de entrevista al arquitecto del subsistema Auditoría de Cedrux, para validar el resultado del modelo matemático.

Entrevista al arquitecto del subsistema Auditoría

Resultado del Modelo matemático para la evaluación de la confiabilidad de Cedrux.

Subsistema: Auditoría									
No	Descripción de los errores detectados	Clasificación	Peso del Error	Camino que recorre	Cálculo de las subcaracterísticas			Confiabilidad	
					M	TC	R	Valor	Interpretación
1	En el componente analizador existen diferentes clases que contienen los mismos métodos debido al mecanismo de integración.	Violación de uso	0.1	E_TC _R	1	1	1	0.10	Confiable
2.	En la interfaz general existe pérdida de etiquetas, debido a la utilización de alertas, y no el mecanismo que define la arquitectura para ser usado.	Especificación incompleta	0.2	E_DA_R	3	2	2	0.47	Confiable
3.	En la clase controladora Exportar existe implementada lógica del negocio.	Violación de uso	0.1	E_TC_R	1	1	1	0.10	Confiable
				Confiabilidad del subsistema:				0.57	Confiable

Resumen e interpretación del resultado

Los errores que están influyendo en la confiabilidad del subsistema evaluado se ajustan a las siguientes clasificaciones:

1. Violación de uso.
2. Especificación incompleta.

Los pesos asignados a estas clasificaciones según su impacto en el sistema son 0.1 y 0.2 respectivamente, lo que determina la superioridad en impacto del error de especificación incompleta, respecto a los de violación de uso. Lo anterior, y el resultado de las confiabilidades iguales de ambos, e inferiores además a la del error de especificación incompleta, ofrecen el orden de importancia de corrección. Sin embargo, al tener estos errores los pesos mínimos de la clasificación, no son capaces de provocar un colapso del sistema.

A partir de los errores de violación de uso, el subsistema toma un estado de tolerancia completa ante el error, es decir no pierde información, ni funcionalidades. Para cada uno de ellos las subcaracterísticas tienen el mejor valor posible (menor), y este puede ser el mejor comportamiento del subsistema respecto a un error por lo que es maduro, tolerante a fallos, y recuperable. No existe la posibilidad de un estado de paro o de fallo total.

El error de especificación incompleta pierde confiabilidad debido a que, a pesar de que sigue funcionando bajo un nivel de ejecución aceptable, el subsistema se degrada perdiendo funcionalidades. La recuperabilidad y la tolerancia a fallos tienen peor confiabilidad que en el caso anterior, sin embargo, la madurez en este caso es la subcaracterística que más afecta al subsistema. Si a partir del error, el subsistema tomara un estado de tolerancia completa, la madurez aumentaría significativamente.

Según lo ya analizado, el subsistema no contiene errores arquitectónicos que puedan hacer que este siga funcionando bajo un nivel de ejecución no aceptable o llegue a un estado de fallo total. El resultado final de la confiabilidad del subsistema es de 0.57, lo que lo ubica según la escala planteada por el modelo en un subsistema **Confiable**.

Pregunta al arquitecto del subsistema Auditoría:

Cuántos años de experiencia tiene en el rol de arquitecto: 4.
En cuántos subsistemas ha desempeñado su rol: 3.
Cuáles: Auditoría, Inventario, CEDRUX.

Nombre: Juan José Rosales Rodríguez
Firma: [signature]

1. De acuerdo a su juicio, y analizando las deducciones obtenidas mediante la ejecución del modelo matemático, coincide usted con la respuesta de este modelo para el subsistema Auditoría:

   √ Si ___No ___Parcialmente.

2. Según su opinión, qué categoría le otorga de manera general a la propuesta:

   √ Excelente ___ Bueno ___ Aceptable ___ Malo

Anexo 10. Modelo de entrevista al arquitecto del subsistema Contabilidad de Cedrux, para validar el resultado del modelo matemático.

Entrevista al arquitecto del subsistema Contabilidad

Resultado del Modelo matemático para la evaluación de la confiabilidad de Cedrux.

Subsistema: Contabilidad										
No	Descripción de los errores detectados	Clasificación	Peso del Error	Camino que recorre	Cálculo de las subcaracteristicas			Confiabilidad	
					M	TC	R	Valor	Interpretación
1.	Se hace uso del fetchAll en vez del query buscando mejorar el rendimiento de la aplicación por la gran cantidad de datos a cargar de otros componentes del sistema	Especificación Incompleta	0.2	E_TC_R	1	1	1	0.20	Confiable
							Confiabilidad del subsistema:	0.20	Confiable

Resumen e interpretación del resultado

El error que está influyendo en la confiabilidad del subsistema evaluado se ajusta a la siguiente clasificación:

1. Especificación Incompleta

El peso asignado a esta clasificación está dado por su poco impacto en el sistema y es de 0.2. El error de especificación incompleta presente, hace que el subsistema tome un estado de tolerancia completa ante el error, es decir no pierde información, ni funcionalidades. Para este error las subcaracterísticas tienen el mejor valor posible (menor), y este puede ser el mejor comportamiento del subsistema respecto a un error por lo que es maduro, tolerante a fallos, y recuperable. No existe la posibilidad de un estado de paro o de fallo total.

Según lo ya analizado, el subsistema no contiene errores arquitectónicos que puedan hacer que pierdan funcionalidades o lleguen a un estado de fallo total. El resultado final de la confiabilidad del subsistema es de 0.20, lo que lo ubica según la escala planteada por el modelo en un subsistema **Confiable**.

Pregunta al arquitecto del subsistema Contabilidad:

Cuántos años de experiencia tiene en el rol de arquitecto: 1.

En cuántos subsistemas ha desempeñado su rol: 1.

Cuáles: Contabilidad.

De acuerdo a su juicio, y analizando las deducciones obtenidas mediante la ejecución del modelo matemático, coincide usted con la respuesta de este modelo para el subsistema Contabilidad:

X Si ___No ___Parcialmente.

Según su opinión, qué categoría le otorga de manera general a la propuesta:

X Excelente ___ Bueno ___ Aceptable ___ Malo

Juan L. Caballero Portellos